AD-A054 471 ARINC RESEARCH CORP ANNAPOLIS MD F/G 15/5
ADAPTATION OF A PROVISIONING MODEL FOR GENERAL-PURPOSE USE BY T--ETC(U)
MAY 70 F J JACOBY, D THOMPSON, B COHEN N00019-70-C-0027
UNCLASSIFIED 928-51-5-1055 NL

1 OF 2
AD
A054471

Publication 539-51-5-1055

FOR FURTHER TRAN

(4)

AD A054471

SPECIAL REPORT NUMBER 5

ADAPTATION OF A PROVISIONING MODEL FOR GENERAL-PURPOSE USE BY THE AVIATION SUPPLY OFFICE

31 May 1970

Prepared for
U. S. NAVY AVIATION SUPPLY OFFICE
PHILADELPHIA, PA.
under Contract N00019-70-C-0027

ARINC RESEARCH CORPORATION

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER 928-51-5-1055 ✓ S.R #5	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) ADAPTATION OF A PROVISIONING MODEL FOR GENERAL-PURPOSE USE BY THE AVIATION SUPPLY OFFICE		5. TYPE OF REPORT & PERIOD COVERED
		6. PERFORMING ORG. REPORT NUMBER 928-51-5-1055
7. AUTHOR(s) F.J. Jacoby, D. Thompson, B. Cohen		8. CONTRACT OR GRANT NUMBER(s) N00019-70-C-0027 new
9. PERFORMING ORGANIZATION NAME AND ADDRESS ARINC Research Corporation ✓, 2551 Riva Road, Annapolis, Maryland 21401		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS U.S. NAVY AVIATION SUPPLY OFFICE, PHILADELPHIA, PA.		12. REPORT DATE May 31, 1970
		13. NUMBER OF PAGES 86
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) U.S. NAVY AVIATION SUPPLY OFFICE, PHILADELPHIA, PA.		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

UNCLASSIFIED/UNLIMITED

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A spares-optimization provisioning model was developed by ARINC Research Corporation for use by the Aviation Supply Office. Given provisioning data for a specific entity such as an aircraft, this model provides the methodology for obtaining an optimum inventory for the entity by using the Poisson distribution. ARINC Research also provided the capability for restructuring Aviation Supply Office data to a format suitable for input to the model.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

(4)

(9)

Special Report, ~~Number 5~~ no. 5

(6)

ADAPTATION OF A PROVISIONING MODEL FOR GENERAL-PURPOSE USE BY THE AVIATION SUPPLY OFFICE

(11) 31 May 1970

(12) 157p.

Prepared for
U.S. Navy Aviation Supply Office
Philadelphia, Pa.
under Contract N00019-70-C-0027

(15)

by

(10)

F. J. Jacoby,
D. Thompson
B. Cohen

ARINC Research Corporation
a Subsidiary of Aeronautical Radio, Inc.
2551 Riva Road,
Annapolis, Md. 21401

Publication 928-51-5-1055

(14)

400 247 LB

ABSTRACT

A spares-optimization provisioning model was developed by ARINC Research Corporation for use by the Aviation Supply Office. Given provisioning data for a specific entity such as an aircraft, this model provides the methodology for obtaining an optimum inventory for the entity by using the Poisson distribution. ARINC Research also provided the capability for restructuring Aviation Supply Office data to a format suitable for input to the model.

ACKNOWLEDGEMENT

ARINC Research Corporation acknowledges the invaluable assistance and cooperation received from the following military and civilian agencies:

- U.S. Navy Aviation Supply Office, Philadelphia, Pa.
  - Systems Planning Division
  - Allowance Control Division
  - Data Processing Division
- Johns Hopkins University, Applied Physics Laboratory, Combat Systems Planning Center, Silver Spring, Md.
- PMA-240, Naval Air Systems Command (NASC), Washington, D.C.

SUMMARY

The spares-optimization model provides a method for obtaining an optimum inventory of spare parts, i.e., an inventory that minimizes backorders at a minimum cost. The two possible program-cutoff constraints are cost and probability of spares adequacy.

Possible sources of provisioning data at the ASO were analyzed, with the final choice being made between the Allowance List File and the Master Data File (MDF). The parameters required to execute the spares-optimization program and the future use of the program at ASO were the primary factors in the decision to use the MDF as the data source. This file provides the most complete and accurate information and is now the established data file for ASO-cognizance items in the UICP (Uniform Inventory Control Point) system.

ARINC Research Corporation developed a computer program that would process data extracted from the MDF in the UICP Input Data Transcript format and structure inputs for the items to be provisioned in a format suitable for the optimization program. This involved creating logic for accurate determination of item quantities, proper handling of failure data that vary because of different item applications (each possibly stressing the item differently), and proper handling of various levels of equipment indenture.

Program narratives, flow charts, listings, and operating instructions were prepared to make the use of the provisioning-model package as smooth as possible. These are presented in this report.

Preceding Page Blank - NOT FILMED

CONTENTS

CONTENTS (continued)

CHAPTER ONE

INTRODUCTION

This report documents the work performed to adapt the ARINC Research spares-optimization model for use by the Aviation Supply Office (ASO) and to provide a guide for proper application of this model. This task, a modification of Naval Air Systems Command (NASC) Contract N00019-70-C-0027, was sponsored by NASC (PMA-240) at the request of the ASO.

Under a previous contract with NASC, ARINC Research Corporation developed a spares-optimization model and applied it to selected subsystems of the P-3C, an anti-submarine warfare (ASW) aircraft being provisioned by the ASO. ASO requested that the NASC contract be modified to include the ARINC Research efforts necessary to adapt the optimization model for use by ASO. To provide more generality and flexibility, several major modifications were necessary to make the program compatible with the procedures and the data-processing system at ASO.

This report describes the technical formulation of the optimization procedure and provides guidance for successful use of the model.

Preceding Page Blank - NOT FILMED

CHAPTER TWO

INVESTIGATION AND ANALYSIS

2.1 DESCRIPTION OF SPARES-OPTIMIZATION MODEL

The spares-optimization model provides a method of obtaining an optimum inventory of spare parts at minimum cost. There are two program-cutoff constaints: (1) cost — i.e., the program will stop purchasing spares when a particular cost constraint is reached, and (2) the probability of spares adequacy obtained by minimizing expected stock back orders (see Appendix A).

Optimization is accomplished by applying an iterative process, which uses the Poisson distribution. The details of the analytic techniques are discussed in Section 2.2.

The output of the probability-constrained optimization program is an initial outfitting list (IOL) and quantities of system stocks (backup stocks). The IOL is an allowance list that indicates the quantities of items to be made available at the time of initial outfitting and to be maintained at a specified activity. These items keep the activity in a material-readiness condition. The system-stocks quantity calculated for each item is the quantity of the item to be maintained at a backup spares location, called the "systems stockage point". This location supports all bases, providing spares for items lost because of wearout and for certain types of items that are being repaired (see Subsection 2.2.3).

The output of the cost-constrained optimization program is the gross spare-parts requirement for a specified distribution of support points as determined by operating plans. The actual level of spares adequacy versus that desired for each base selected is summarized, as is cost.

2.2 ANALYTIC TECHNIQUES

2.2.1 Introduction

The ARINC Research spares-optimization model was originally tailored to fit the maintenance philosophy for the P-3C aircraft, and the optimizing technique used constraint values peculiar to the P-3C. To give the ASO the capability of applying this same procedure, and to provide more generality and flexibility, the primary equations of the model were modified through the joint efforts of ASO personnel in the Allowance Control and Systems Planning Divisions and ARINC Research personnel. The modifications permit many different provisioning situations to be handled simply by changing variables associated with maintenance-philosophy determination.

2.2.2 Types of Maintenance Locations

Two general types of maintenance locations must be considered:

1. The operational base at which organizational-level maintenance is carried on as well as intermediate-level maintenance (IMA)
2. The depot or Overhaul and Repair (O&R) activity

There are two types of stockage locations that correspond to these maintenance locations:

1. The base supply stocks (maintained at the operational base)
2. Systems stocks or backup spares stocks (maintained at the depot or O&R activity)

2.2.3 Repair Categories

Items to be provisioned are categorized in five categories according to their repairability — i.e., whether they are repairable or consumable (throw-away types) — and the locations at which they are repaired or thrown away:

1. Depot Repairable — an item that can be repaired only at the depot or O&R activity
2. Base Repairable — an item that can be repaired at the operational base (this category includes items repaired at the organizational level as well as those that undergo intermediate-level repair)
3. Base/Depot Repairable — an item that is repaired a certain percentage of the time at the operating base and the remainder of the time at the depot or O&R activity
4. Base Consumable — an item of the throw-away type that is replaced and discarded at an operating base
5. Depot Consumable — an item of the throw-away type that is replaced and discarded at the depot or O&R activity

2.2.4 Data Required

The average-demand equations, discussed in Subsection 2.2.5, require the data elements summarized in Table 1. This table includes the abbreviation or symbol for the data element, the dimension of the element, the equation number in which the element is used, and a brief description of the element.

2.2.5 Average-Demand Equations

The equations presented in this subsection are average-demand formulas for spares for each of the item types described in Subsection 2.2.3. The demands determined by these formulas are used in the spares-optimization iterative process, which employs the single-parameter Poisson distribution. This process is described in Subsection 2.2.6.

2.2.5.1 Spares Required for Operating Bases

The average-demand equations presented in this discussion are used to develop the IOL. No average-demand equation is presented for depot-consumable items, since these spares are provided only to the systems stockage point.

Table 1. **ELEMENTS USED IN THE AVERAGE-DEMAND EQUATIONS**

Name	Symbol	Dimension	Equation Number	Description
Flying hours per month	$\frac{FH}{M}$	$\frac{\text{Hours}}{\text{Month}}$	1, 2, 3, 4	A value assigned for each allowance-list column representative of various flying-hour programs.
Flying hours per month for consumable and wear-out items	$\frac{FH}{Mcw}$	$\frac{\text{Hours}}{\text{Month}}$	5, 7, 8, 9, 10	A value representing the average value of flying hours per month considering the aircraft-production schedule for the requisitioning objective.
Flying hours per month for repairable items	$\frac{FH}{M_r}$	$\frac{\text{Hours}}{\text{Month}}$	6, 7, 9	A value representing the average value of flying hours per month for the period representing the difference between the requisitioning objective and the recovery maintenance-cycle period.
Turn-Around-Time IMA	TAT	Days	2, 3	The time, in days, required to remove a failed item from the aircraft, ship it to the intermediate maintenance activity, and return it to the base stockage point.
Resupply Time	RT	Days	1, 3	The time, in days, required to receive an item at the base from the systems stockage point following the placing of a requisition due to a removal and possible failure of an item from an aircraft.
Protection Time	PT	Days	4	The period, in days, for which a base requires a stock of a consumable item.
Restockage Time	RST	Days	7, 9	The time, in days, to remove an item from an aircraft, ship it to the depot, repair it, and send it in ready-for-issue condition to the systems stockage point.
Rotable Pool Factor	RPF	$\frac{\text{Removals}}{\text{One Maintenance Cycle}}$	2, 3, 8	The number of times a repairable assembly will be removed from an aircraft and repaired at an intermediate level of maintenance or below in one maintenance cycle.
Maintenance Cycle	MC	Hours	1, 2, 3, 4, 5, 6, 7, 8, 9, 10	A base established for computing spare-parts requirements. One maintenance cycle is equal to 100 flying hours.
Maintenance Replacement Factor	MRF	$\frac{\text{Removals}}{\text{One Maintenance Cycle}}$	1, 3, 4, 7, 9, 10	For a consumable item, the number of times the item will require replacement in an aircraft or equipment in one maintenance cycle. For a repairable assembly, the number of times an assembly will be beyond the repair capability of the IMA in one maintenance cycle.

(continued)

Table 1. (continued)				
Name	**Symbol**	**Dimension**	**Equation Number**	**Description**
Quantity per Application	QA	Dimensionless	1, 2, 3, 4, 7, 8, 9, 10	ASO definition: "A numerical expression of the quantity of a specific item in a specific higher entity; e.g., quantity per assembly, component, equipment, or end article."
Number of Higher Applications	AI	Dimensionless	1, 2, 3, 4 7, 8, 9, 10	The total number of the specific higher entity in which the specific item is contained.
Percent per Application	PA	Dimensionless	7, 8, 9, 10	ASO definition: "A percentage expression of the total application population to which the item applies."
Overhaul Replacement Rate	OR	Dimensionless	10	ASO definition: "A decimal rate assigned to an item to cite the provisioning estimate of the anticipated requirement for the item for use in Overhaul or Repair of a particular application at the Depot level."
Rework Removal Rate	RRR	Dimensionless	7, 8, 9	The anticipated percentage of the total quantity of a repairable assembly on an aircraft or engine passing through the overhaul and repair that will require some depth of rework.
Wearout Rate	Z	Dimensionless	7, 8, 9	ASO definition: "A decimal rate which represents the percentage *of repairable items that fail,* which will not, through rework, be returned to serviceable condition."
Next-Higher-Assembly Overhauls	NHA OHLS	Dimensionless	7, 8, 9, 10	The number of overhauls of the next higher assembly in which an item is contained.
Contract Production-Lead-Time Average	PL	Quarters	7, 8, 9, 10	ASO definition: "The number of months covering the time interval between placement of the contract and the end of the first month in which shipments less expedites has equaled the monthly issue rate plus one month; or the number of months covering the time interval between placement of a contract and shipment into the Supply System of 25% of contracted quantity plus one month, whichever occurs first."

Depot-Repairable Items

As shown in Figure 1, the depot-repairable item is shipped directly from the operating base to the depot and the system stockage point provides the base supply stocks with an operational replacement. After repair, the original item is sent from the depot to the systems stockage point for eventual reissue. There must be sufficient stocks at the base supply point to protect the base from "stockout" during the interval in which items are being shipped from the systems stockage point. This interval is called 'resupply time' (RT).

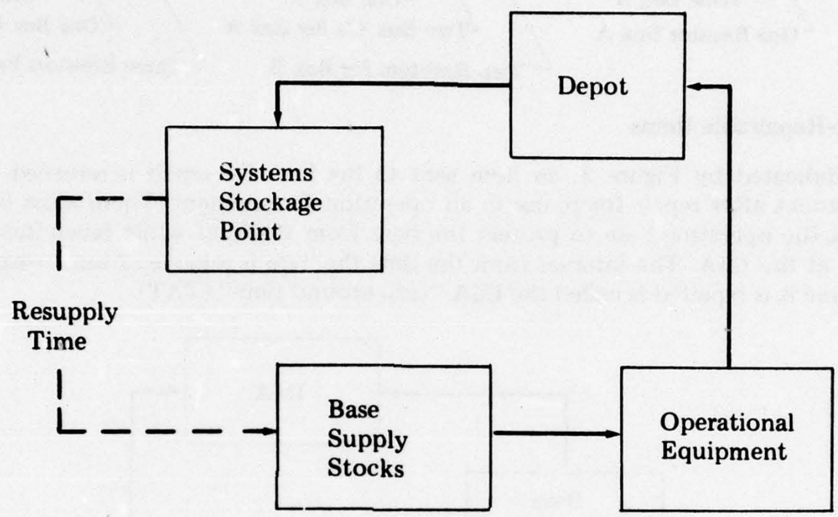


Figure 1. **DEPOT REPAIR LOOP SHOWING RESUPPLY TIME**

The average-demand equation for depot-repairable items is as follows:

$$\text{Average Demand} = \left(\frac{FH}{M}\right)\left(\frac{RT}{30}\right)\left(\frac{MC}{HRS}\right)\left[\sum_{m=1}^{w}\left(MRF_m\right)\left(QA_m\right)\left(AI_m\right)\right] \quad (1)$$

where w equals number of applications. The summation expression of Equation 1 is common to all the average-demand equations.

The use of this type of expression can be explained best by the example shown in Figure 2. Box A, a repairable assembly, contains three repairable subassemblies — two Box Bs and one Box C. Each of the boxes contains a number of resistors, R; the resistors, by

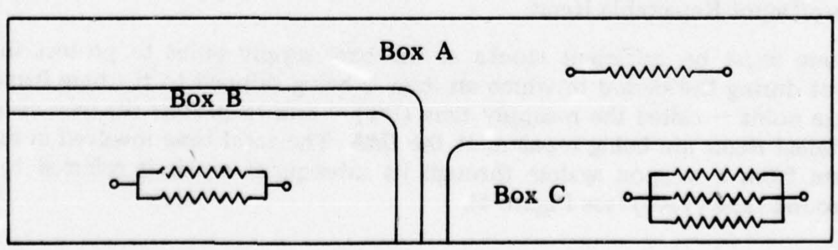


Figure 2. **SUMMATION EXAMPLE**

virtue of their use in the three boxes, have three different MRFs. To express the demand for this item, it is necessary to consider a weighted value of MRF and the quantity per each higher-level application, as follows:

$$\sum_{m=1}^{w} \left(MRF_m\right) \left(QA_m\right) \left(AI_m\right) =$$

$$\left(MRF_A\right) \underbrace{\left(1\right)}_{\text{One Resistor Box A}} \underbrace{\left(1\right)}_{\text{One Box A}} + \left(MRF_B\right) \underbrace{\left(2\right)}_{\text{Two Resistors Per Box B}} \left(\underbrace{2}_{\text{Two Box B's Per Box A}} \times \underbrace{1}_{\text{One Box A}}\right) + \left(MRF_C\right) \underbrace{\left(3\right)}_{\text{Three Resistors Per Box C}} \left(\underbrace{1}_{\text{One Box C Per Box A}} \times \underbrace{1}_{\text{One Box A}}\right)$$

Base-Repairable Items

As indicated by Figure 3, an item sent to the IMA for repair is returned to the base supply stocks after repair for re-use in an operational equipment. There must be sufficient stocks at the operating base to protect the base from stockout while failed items are being repaired at the IMA. The interval from the time the item is removed from a weapon system to the time it is repaired is called the IMA "turn-around time" (TAT).

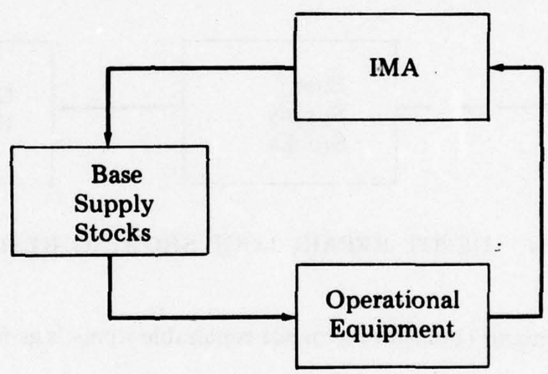


Figure 3. **IMA REPAIR LOOP**

The average-demand equation for base-repairable items is as follows:

$$\text{Average Demand} = \left(\frac{FH}{M}\right) \left(\frac{TAT}{30}\right) \left(\frac{MC}{HRS}\right) \left[\sum_{m=1}^{w} \left(RPF_m\right) \left(QA_m\right) \left(AI_m\right)\right] \qquad (2)$$

Base/Depot-Repairable Items

There must be sufficient stocks at the base supply point to protect the base from stockout during the period in which an item is being shipped to the base from the systems stockage point — called the resupply time (RT) — and to protect the base against stockout while failed items are being repaired at the IMA. The total time involved in the removal of the item from a weapon system through its subsequent repair is referred to as the IMA turn-around time (TAT) (see Figure 4).

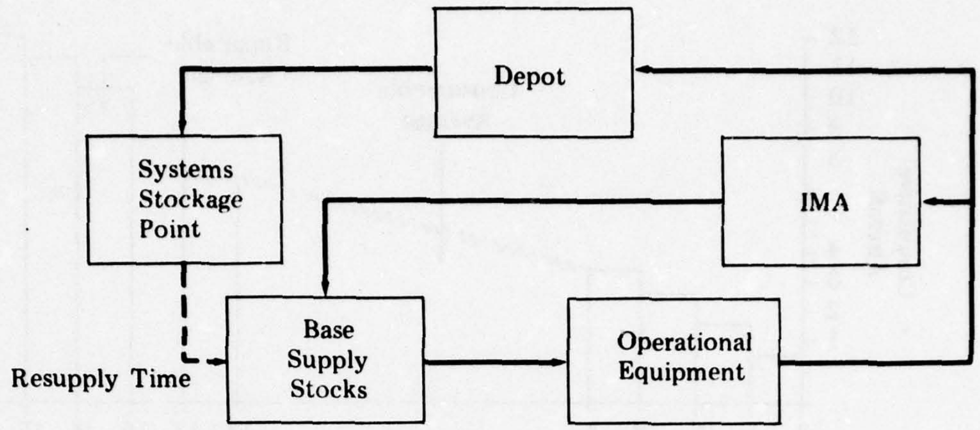


Figure 4. **BASE DEPOT REPAIR LOOP**

The average-demand equation for base/depot-repairable items is as follows:

$$\text{Average Demand} = \left(\frac{FH}{M}\right)\left(\frac{MC}{HRS}\right)\left[\sum_{m=1}^{W}\left(RPF_m\right)\left(QA_m\right)\left(AI_m\right)\left(\frac{TAT}{30}\right) + \sum_{m=1}^{W}\left(MRF_m\right)\left(QA_m\right)\left(AI_m\right)\left(\frac{RT}{30}\right)\right] \qquad (3)$$

Base-Consumable Items

The variable protection time (PT) specifies the number of days' stock for base-consumable items desired at the operating base. The average-demand equation for base-consumable items is as follows:

$$\text{Average Demand} = \left(\frac{FH}{M}\right)\left(\frac{PT}{30}\right)\left(\frac{MC}{HRS}\right)\left[\sum_{m=1}^{W}\left(MRF_m\right)\left(QA_m\right)\left(AI_m\right)\right] \qquad (4)$$

2.2.5.2 Spares Required for Systems Stockage Points

This discussion presents average-demand equations for determining the stocks of items required at systems stockage points.

After the initial provisioning of the operating bases, the stockage points become the sources of parts for the bases. The equations will yield the average quantity of spares of an item that must be stocked at the systems stockage point to support the bases for the production lead time of a particular item.

Before considering the equations for average demand at the systems stockage point, a method for determining the flying hours per month (FH/M_{cw} and FH/M_r) should be discussed. A normal graph of the cumulative number of operating aircraft versus time for a new weapon system is a step function like that shown in Figure 5.

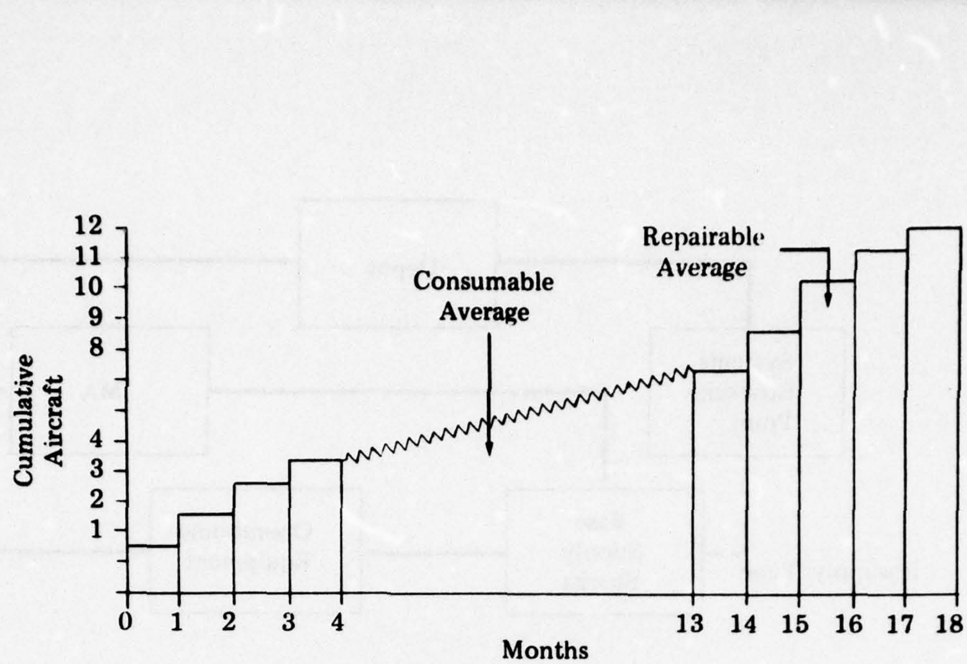


Figure 5. **PRODUCTION-SCHEDULE GRAPH**

In the case of a weapon-system program in which the number of aircraft supported per month is not a constant, it is necessary to select an appropriate value of flying hours per month to be used in calculating average demand in the equation for system-backup stocks. For consumable items and items that are expected to be lost because of repairable-item wearout, this is not a particularly difficult problem. The average flying hours per month are determined on the basis that the entire requisitioning objective (RO) (the period of time considered for spares support) must be supported.

How this average is determined can be seen by examining Figure 5 and considering the following:

> As each month passes a number of aircraft are added to the total inventory. Each month the new cumulative number of aircraft is considered in determining an incremental area under the curve. The value representing this area, expressed in aircraft months, is multiplied by that month's flying-hour program (FH/M), resulting in average flying hours for that month. The sum of these figures for each month in the requisitioning objective represents the total average flying hours. Dividing this average by the number of months in the requisitioning objective gives average flying hours per month.

The average flying hours per month for consumables as calculated above is multiplied by the production lead time (in months) of a specific item to give an average number of flying hours to be used in the average-demand equation for that item.

For repairable assemblies a similar approach can be used. Since the average flying hours per month for repairables should be representative of the number of aircraft from which items are being placed in the pipeline, the use of a simple average is not satisfactory because it would leave the stockage point short of spares in the latter portion of the requisitioning objective. To compensate, an *average is taken over the last five months* of the requisitioning objective. The use of this average value in the average-demand equation ensures that sufficient spares will be generated.

The following equations can be used to determine average flying hours per month:

- Consumables and items lost due to wearout —

$$\frac{FH}{M_{cw}} = \left(\frac{HRS}{MC}\right)\left(\frac{\text{MC in requisitioning objective}}{RO}\right) \tag{5}$$

- Repairables —

$$\frac{FH}{M_r} = \frac{\left(\frac{HRS}{MC}\right)\left[\text{MC in last 5 months of requisitioning objective}\right]}{\text{5 (months)}} \tag{6}$$

Equations 5 and 6 must be hand-calculated and the results entered as initialization parameters to the program.

Depot-Repairable Items

The average-demand equation for depot repairables at the systems stockage point is written as follows:

$$\text{Average Demand} = \left(\frac{MC}{HRS}\right)\left[\sum_{m=1}^{w}\left(MRF_m\right)\left(QA_m\right)\left(PA_m\right)\left(AI_m\right)\right]\left[\left(\frac{FH}{M_r}\right)\left(\frac{RST}{30}\right) + \left(\frac{FH}{M_{cw}}\right)(PL)(Z)(3)\right] + \sum_{m=1}^{w}\left(OHLS\ NHA_m\right)\left(RRR_m\right)\left(QA_m\right)\left(PA_m\right)(Z) \tag{7}$$

This equation is designed to allow stockage of spares to protect the stockage point from stockout during the period in which items are being shipped to and being repaired at the depot, to provide spares for repairable items lost because of wearout, and to provide spares to meet an additional requirement for spares support of overhauls (see Figure 6).

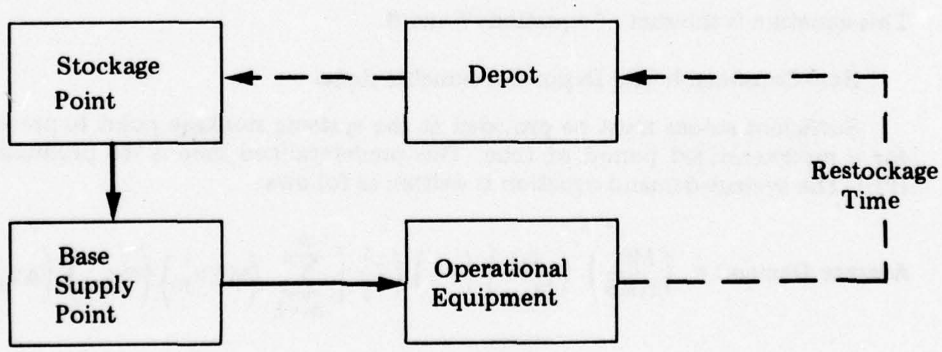


Figure 6. **DEPOT REPAIR LOOP SHOWING RESTOCKAGE TIME**

Base-Repairable Items

The average-demand equation for base repairables at the systems stockage point is as follows:

$$\text{Average Demand} = \left(\frac{FH}{M_{cw}}\right)(Z)(PL)(3)\left(\frac{MC}{HRS}\right)\left[\sum_{m=1}^{w}(RPF_m)(QA_m)(PA_m)(AI_m)\right] + \sum_{k=1}^{w}(OHLS\ NHA_k)(RRR_k)(QA_k)(Z)(PA_k) \tag{8}$$

Basically, the equation is designed to provide for spares to be stocked to allow replacement of repairable items lost because of wearout, plus an additional quantity for items required because of overhauls.

Base/Depot-Repairable Items

The average-demand equation for base/depot repairables at the systems stockage point is as follows:

$$\begin{aligned}\text{Average Demand} = {} & \left(\frac{MC}{HRS}\right)\left[\sum_{m=1}^{w}(MRF_m)(QA_m)(PA_m)(AI_m)\right]\left[\left(\frac{FH}{M_r}\right)\left(\frac{RST}{30}\right)\right. \\ & \left. + \left(\frac{FH}{M_{cw}}\right)(PL)(Z)(3)\right] + \left(\frac{FH}{M_{cw}}\right)(Z)(PL)\left(\frac{MC}{HRS}\right)\left[\sum_{m=1}^{w}(RPF_m)(QA_m)\right. \\ & \left.(PA_m)(AI_m)(3)\right] + \sum_{m=1}^{w}(OHLS\ NHA_m)(RRR_m)(QA_m)(PA_m)(Z)\end{aligned} \tag{9}$$

This equation is the sum of equations 7 and 8.

Base-Consumable and Depot-Consumable Items

Sufficient stocks must be provided at the systems stockage point to preclude stockout for a predetermined period of time. This predetermined time is the production lead time (PL). The average-demand equation is written as follows:

$$\text{Average Demand} = \left(\frac{MC}{HRS}\right)\left(\frac{FH}{M_{cw}}\right)(PL)(3)\left[\sum_{m=1}^{w}(MRF_m)(QA_m)(AI_m)(PA_m)\right] + \frac{(PL)(3)}{RO}\left[\sum_{m=1}^{w}(OHLS\ NHA_m)(OR_m)(QA_m)(PA_m)\right] \tag{10}$$

2.2.6 Elements of the Optimization Procedure

The general calculations and procedures involved in the operation of the spares-optimization model can be outlined as follows and as shown in Figure 7:

1. The average demand (AD) for each item that is to be considered in a particular provisioning is calculated in the manner described in Subsection 2.2.5.
2. With an inventory level initially set to zero, a calculation is made for each of "f" items to determine the reduction in back orders that would be obtained by adding one spare to the inventory. For each item, this can be expressed as

$$BR = E\,[B(N_i - 1)] - E\,[B(N_i)] = 1 - \sum_{m=0}^{N_i - 1} \frac{e^{-AD_i}\,(AD_i)^m}{m!} \tag{11}$$

3. For each item, the value representing reduction in expected back orders is divided by the item's unit cost. This will result in "f" values of need-cost factors, expressed as

$$\text{Need Cost Factor (i)} = \frac{BR}{C(i)} \tag{12}$$

4. The item that has the highest need-cost factor is selected and assigned one spare, and the amount it costs is considered expended.
5. The total spent for spares is compared with a cost constraint if this comparison is desired. If the amount expended exceeds the cost constraint, the program stops.
6. The probability of spares sufficiency for a particular item is expressed by the cumulative Poisson distribution as

$$P\,[AD_i \leq (N_i - 1)] = \sum_{m=0}^{N_i - 1} \frac{e^{-AD_i}\,(AD_i)^m}{m!} \tag{13}$$

   The overall probability of sufficiency is obtained by multiplying the individual item probabilities together. This can be expressed as

$$\prod_{i=1}^{f} P\,[AD_i \leq N_i - 1] \tag{14}$$

   If the probability constraint is desired and is satisfied — i.e., the product exceeds an entered constraint value — the program stops.
7. For the item for which a spare was purchased, the reduction in expected back orders again is calculated with N incremented by one. In addition, its new need-cost factor is calculated. The procedure is then repeated as outlined above, starting with Step 4.

Techniques for reducing the time required to perform the foregoing procedure have been implemented in the program developed for ASO. They involve setting the iterative-process starting point to a value that will minimize the number of required iterations.

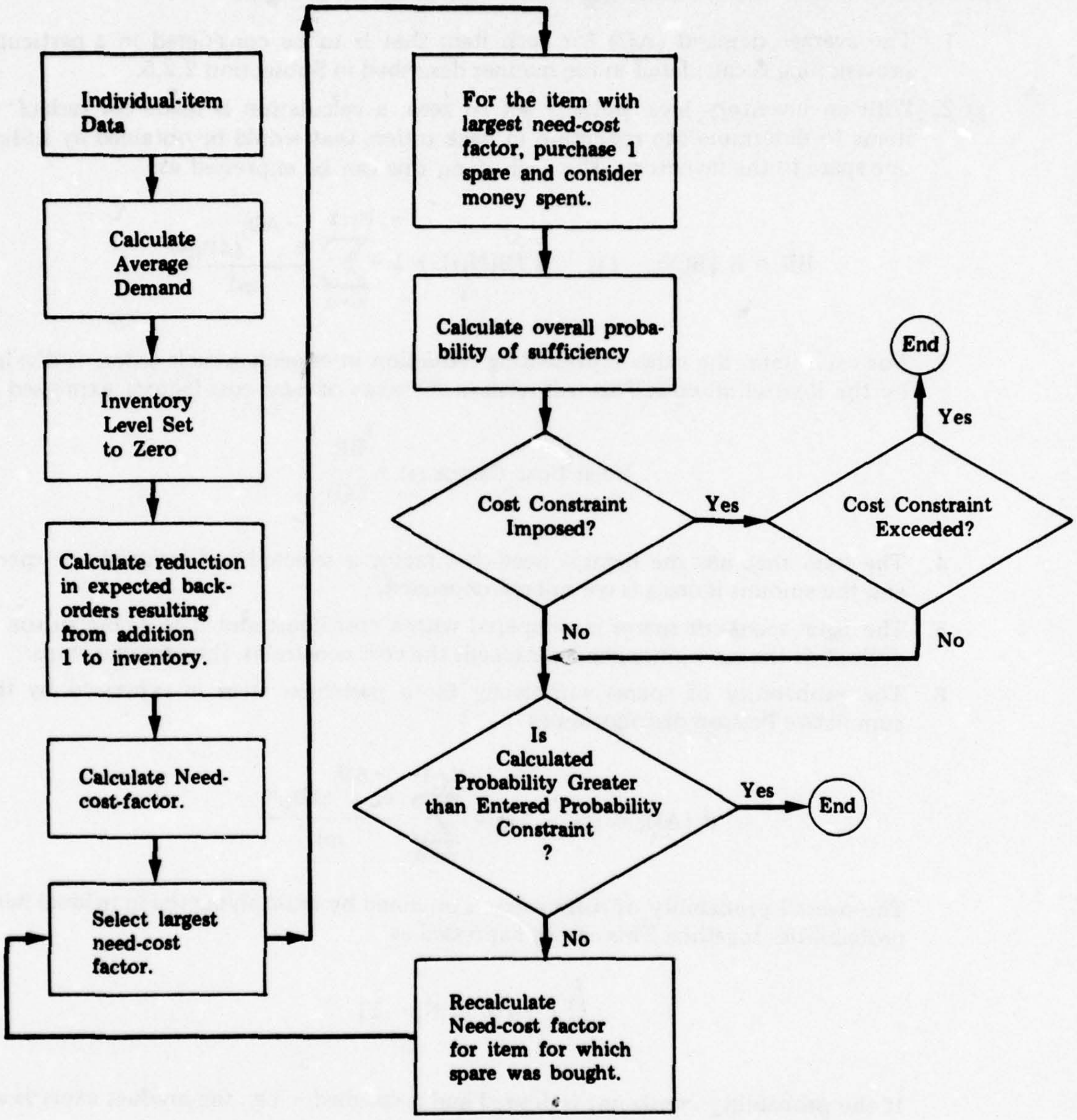


Figure 7. **GENERAL FLOW OF SPARES-OPTIMIZATION PROGRAM**

CHAPTER THREE

ANALYSIS OF ASO FILES

To make the spares-optimization provisioning model compatible with ASO data-processing files, ARINC Research Corporation analyzed the structure and content of these files and determined that the two best choices for data sources were the Allowance List File and the Master Data File. This decision was based primarily on the fact that these files contain required data in a reasonably usable form.

3.1 ANALYSIS OF MASTER DATA FILE (MDF) DOCUMENTATION

ASO Instruction P4440.60A, dated 1 July 1969, Subject: Files Maintenance Under UADPS-ICP (Uniform Automated Data Processing Systems-Inventory Control Point), was reviewed. The data elements being maintained in the MDF that would be directly applicable as inputs to the spares-optimization program were identified. They are listed in Table 2.

3.2 ANALYSIS OF MASTER DATA FILE UPDATE

The MDF is maintained by the Univac 490 series system of computers, which permits the entire file to be accessed randomly. The file thus can be updated or changed by using data element numbers (DENs) as references without resorting to complete record replacement or extensive data manipulation. Allowance-list spreads for use in an IOL are entered by such an updating process, using DENs DØØ5/CØØ7. The MDF then can be used as a source to update the tape records referred to as the Allowance List Files.

3.3 ANALYSIS OF ALLOWANCE LIST FILE

Review of the Consolidated Aviation Allowance List Transcript 4ND-ASO-4441/17 as well as the latest master-tape format of the Allowance List File records maintained by the Data Processing and Allowance Control Divisions revealed that the data elements required for use by the spares-optimization program were quite incomplete. Allowance-list spreads can be updated in the Allowance List File by a file interface with the MDF.

3.4 FILE SELECTION AND RESULTANT ACTION

The relationship of highest-level repairable assemblies to lower-level nested repairables ("sons", "grandsons", etc.), as well as the application of piece parts to these repairables, is very complex. The data-file source that would be most helpful in determining these relationships was found to be the MDF. Therefore, ARINC Research selected that file as the most

Table 2. **DATA ELEMENT NUMBERS (DEN'S) USED BY DATA-CONVERSION PROGRAM**

Data Element Number (DEN)	Title
BØØ2	Local Routing Code (LRC)
BØ1Ø	Contract Production Lead Time Average
BØ53	Unit Price
BØ55	Unit Price, Item Replacement
BØ67	Rules Code
CØØ1E	NATO Country Code
CØØ2	Activity Control Number
CØØ3	Cognizance Symbol
CØØ4	Item Name
CØØ5	Unit of Issue
CØ35	Federal Supply Code for Manufacturers (FSCM)
CØ42	Federal Supply Classification
DØØ1	Reference Number
DØØ8	Repairable Identification Code — RIC (Model Code)
DØ46	Federal Item Identification Number (FIIN)
DØØ9	Application Code
DØ11	Quantity Per Application
DØ12	Source Code
DØ13	Maintenance Code
DØ13C	Maintenance Condemnation Code
DØ29	Application/Identification Number Activity Code (AINAC)
EØØ7	Provisioning Insurance Quantity
FØØ1	Maintenance Replacement Rate
FØØ3	Overhaul Replacement Rate
FØØ7	Wearout Rate
FØ18	Percent Per Application
DØØ5/CØØ7	Allowance List Quantities

logical data source for the optimization programs. Other factors that influenced this selection were the random-access update feature, the intention of the ASO to use the MDF for allowance-list maintenance in the future, and the completeness of MDF data entries.

A Data Conversion Program (see Appendix B) was written to generate the key parameters for the optimization programs (see Appendixes C and D) by using the MDF data as inputs.

3.5 ANALYSIS AND USE OF UICP DATA CARD FORMAT

The spares-optimization program is designed to use provisioning data in the UICP Input Data Transcript format (4ND-ASO-4423/45A/B/C). These data can be made available to the program user in two ways. First, an input tape, generated by ASO in the UICP Data format and containing selected data element numbers (DENs) extracted from the MDF, can be used as input to the ARINC Research data-conversion program. This is a preferred method because the information thus obtained will be complete and well edited. The second method, which is less desirable, is to use data cards in the UICP Data format, punched before the information is entered into the MDF.

One drawback of using data cards is that if any information pertaining to a particular item was previously entered into the MDF, a card would not be generated for that piece of information a second time. Consequently, information on long-lead items, for example, might not be included in the package of data cards desired for use with the optimization process. This would result in an incomplete optimization of the provisioning.

The data-conversion program uses the UICP data to structure an input suitable for the optimization program. The data-conversion process is described in the program narrative, Appendix B.

Preceding Page Blank - Not Filmed

CHAPTER FOUR

DEMAND-FLOOR OPTION

One of the many problems faced by the ASO is that an excessive range of items is being carried currently on the IOLs. To reduce the quantity of items being treated by the optimization process without jeopardizing spares sufficiency, several approaches to range reduction were considered.

A method called the "demand-floor option" was selected; this allows the program user to eliminate those items that, according to a specific maintenance philosophy or in his own judgement, do not warrant consideration in the optimization process (i.e., their demand over a specified period is less than a reference value, called the "demand floor"). This elimination is achieved by providing the program with test parameters for each type of item. A sample test parameter is "1,6" — which represents one demand in six months. In this case, if an item has less than one demand in six months, it is not considered in the optimization process and zero spares are assigned in the IOL. Any level of range reduction desired can be obtained by providing appropriate parameters. The demand per month is derived from the average demand equations (see Appendix C).

Preceding Page Blank - NOT FILMED

CHAPTER FIVE

SUBSYSTEM ALLOCATION OF FUNDS

A problem commonly encountered in the provisioning of a specific weapon system is the relationship between budgeted spares funds for the weapon system and the incremental provisioning of the subsystems of the weapon system. When the incremental subsystem provisionings take place, the question arises as to how much of the total weapon-system budget to spend on the subsystem being provisioned.

In one *a priori* technique for determining the amount of money to be allocated to each subsystem of a weapon system, the various subsystems or groups of subsystems that will undergo separate provisionings are identified and grouped together. Then estimates are made for all pertinent data required to run the spares-optimization program on an item-by-item basis. These estimates need not be made for all items in a particular provisioning; however, the accuracy of the final answer will correlate roughly with the accuracy of the estimates made and the number of items covered. When all items are not included in the original estimate, it is important that the estimates start with the highest-ranked item and work downward. The ranking is accomplished by multiplying item cost by item failure rate.

Once these estimates have been made, the spares-optimization program is initialized for the probability run, and an arbitrary or estimate IOL is developed for each separate subsystem being provisioned.

Finally, the operational planning data are used to determine a gross requirement for spares, which is used as a baseline for developing the apportionment as described in the following example.

Suppose that weapon system XYZ has available for a particular fiscal year's provisioning $20 million of PAMN funds. Further suppose that the subsystems of the weapon system will be provisioned in five separate provisionings and that an arbitrary IOL has been developed for the five groups of equipment. Suppose that the operational planning data consist of a column-8 IOL selection and the system backup; then the following numbers are calculated for the five groups of equipment:

	Column 8 + System Backup Cost ($ Millions)
	5
	10
	1
	2
	4
Total:	22

Then the apportionment of the $20 million budget would be as shown in Table 3.

Table 3. **APPORTIONMENT OF SAMPLE BUDGET**

System	Apportionment Factor	Total Budget ($Millions)	Apportioned Budget ($Millions)
1	5/22	20	$\frac{50}{11}$
2	10/22	20	$\frac{100}{11}$
3	1/22	20	$\frac{10}{11}$
4	2/22	20	$\frac{20}{11}$
5	4/22	20	$\frac{40}{11}$
			Total = $\frac{220}{11}$ = $20 Million

CHAPTER SIX

CONCLUSIONS AND RECOMMENDATIONS

6.1 *CONCLUSIONS*

The following conclusions were reached as a result of ARINC Research's efforts in adapting a provisioning model for the Aviation Supply Office:

- The provisioning model, as adapted, can be used by the Aviation Supply Office. Only items in the UICP Input Data Transcript Format (4ND-ASO-4423/45) will be handled by the provisioning model.
- The Master Data File (MDF) is the most appropriate source of information for use as inputs to the provisioning model.
- The MDF data must be restructured to be used by the provisioning model; this was the reason for developing the data-conversion program.

6.2 RECOMMENDATIONS

The following recommendations are made:

- Currently, the Master Data File (MDF) contains information only on items under the cognizance of the ASO. It is recommended that extraction programs, with output format the same as the MDF extraction format, be written for files containing information on items not under ASO cognizance. This would allow the handling of complete provisionings in one program operation.
- The development of compatible models for use with the ARINC Research model, such as an inventory model and a replenishment model, should be considered.
- Initialization parameters provided by the ASO managers should be coordinated by an analyst to check for consistency and to provide the data-processing division with a single package for creation of the data cards.
- A portion of the Master Data File should be reserved to keep critical information intact for an in-process provisioning.

APPENDIX A

DEFINITION OF 'OPTIMUM' AND MATHEMATICAL ASSUMPTIONS

This appendix presents a more detailed definition of "optimum", gives the necessary assumptions used in the optimization methodology, and shows the validity of the assumptions for the optimization procedure.

"Optimum", as used in the ASO provisioning models, is defined as that inventory obtained by minimizing expected back orders, for a minimum cost, within a specified probability constraint, or by minimizing expected back orders for a given dollar cost.

The expected number of back orders (unfilled demands) for a part with N_i spares is given by:

$$E\,[B(N_i)] = \sum_{k=N_i+1}^{\infty} (k-N_i)\, P_i(k)$$

If the number of spares for part i is increased from N_i to $N_i + 1$, the reduction in back orders per additional dollar spent is

$$\frac{E\,[B(N_i)]\ -\ E\,[B(N_i+1)]}{C_i} = \frac{1-\sum_{k=0}^{N_i} P_i\,(k)}{C_i}$$

The optimization procedure for choosing the spares assignment is stepwise. The first item for which one spare is chosen is that for which

$$\frac{1}{C_i}\,[1-\sum_{k=0}^{N_i} P_i\,(k)]$$

is maximum. If this is the j^{th} part, then $N_j = 1$ and all other N_i, $i \neq j$ values remain zero. If the constraint is not violated, the procedure is repeated so that when the current spares assignment is $(N_1, N_2 \ldots, N_m)$, a particular cross-sect.on of spares in time, a spare is always added for the part for which

$$\frac{1}{C_i}\,[1-\sum_{k=0}^{N_i} P_i\,(k)]$$

is maximum.

This procedure is based on the economic principle of marginal or incremental analysis. In this case, we consider the ratio of the incremental decrease in expected back orders to the incremental increase in cost.

To implement this approach, an equation for P(k) is necessary. The model uses the well known result of Palm that if demands are Poisson-distributed with rate λ and mean repair or resupply time is T, the number in repair or resupply in the steady state is Poisson with parameter λT; therefore, if a Poisson demand is assumed (equivalent to a constant failure rate), we have

$$P_i(k) = \frac{e^{-\lambda_i T_i} \left(\lambda_i T_i\right)^k}{k!}$$

Optimization based on expected back orders is theoretically acceptable since several assumptions can be reasonably made. The first of these concerns the assumption of a constant arrival or demand rate. It is not unreasonable to expect that if one or several of a squadron's planes are unavailable, then the remaining planes would take up the slack by flying a greater number of hours. When many planes are experiencing shortages, however, then this assumption may not be reasonable. However, since the sparing procedure will yield high availabilities, a large number of plane shortages is unlikely and the optimization process will yield results that are consistent with the stated goals.

The second major assumption (which actually can also be used to justify a constant demand rate) is that some type of emergency procedure exists so that necessary parts can be obtained when plane availability reaches a critical stage.

For example, if a squadron consists of ten planes, the procedure might be such to obtain immediately, through some special source, the necessary parts to maintain at least six available planes. These parts then become part of the inventory, and by this procedure the use of a theoretically infinite number of back orders becomes justifiable.

APPENDIX B

DATA-CONVERSION PROGRAM

To provision spare parts for an item contained within a specific entity (using the spares-optimization program), entity being defined as an aircraft, system, etc., it is necessary to know certain attributes of the item. These attributes include the true quantity of the item in the entity, as well as an accurate representation of the item's failure characteristics derived from consideration of the maintenance replacement factors, rotable-pool factors, and overhaul replacement rates.

To determine the attributes, it was necessary to develop the data-conversion program. The data-conversion program processes input data in the UICP Data Card Format as shown in Figure B—1. These processed data are then inputted for use in the average-demand equations in the spares-optimization program.

1. PASS 1

The data-conversion program developed by ARINC Research Corporation is designed to restructure data from the UICP Input Data Transcript format into a form suitable for input to the spares-optimization model (see Figure B—1). The input to the data-conversion program is a tape supplied by ASO that includes the information (DENs) for each FIIN that is required by the spares-optimization program. The first part of Pass 1 of the data-conversion program consists of an edit-and-sort routine. This sub-program deletes any DENs not necessary for consideration by the data-conversion program and then sorts the DENs remaining for each FIIN by the following criteria: all DENs other than DEN DØØ9 are carried in the order read; all DØØ9s come last, with DØ29 sub-DENs occurring in the first DØØ9 string. These DENs are entered on a drum set for later retrieval.

In the data-conversion program, the records are read sequentially and each record is processed in turn. In essence, the technique employed is to identify the DEN in each record; then, by use of the reread feature available in UNIVAC 490 FORTRAN, the input buffer is read again with the format required for retrieving the data corresponding to this particular DEN.

Pertinent descriptive information is retrieved and saved; this includes the nomenclature, federal supply code for manufacturers, federal supply class, unit of issue, etc. When this portion of the program has been completed for a particular FIIN, all remaining DENs for this FIIN are DØØ9s; the program then begins to test for DØØ9 sub-DENs. All DØ29s are processed first, and the application codes corresponding to the DØ29s are stored in arrays that correspond to the application/identification number activity code (AE, AT, AR, AP, AQ, or AC); these application codes are later used as references in processing the remainder of the sub-DENs. The application activity codes indicate that the specific application is to an aircraft, is a highest-level repairable assembly, is a nested lower-level repairable assembly,

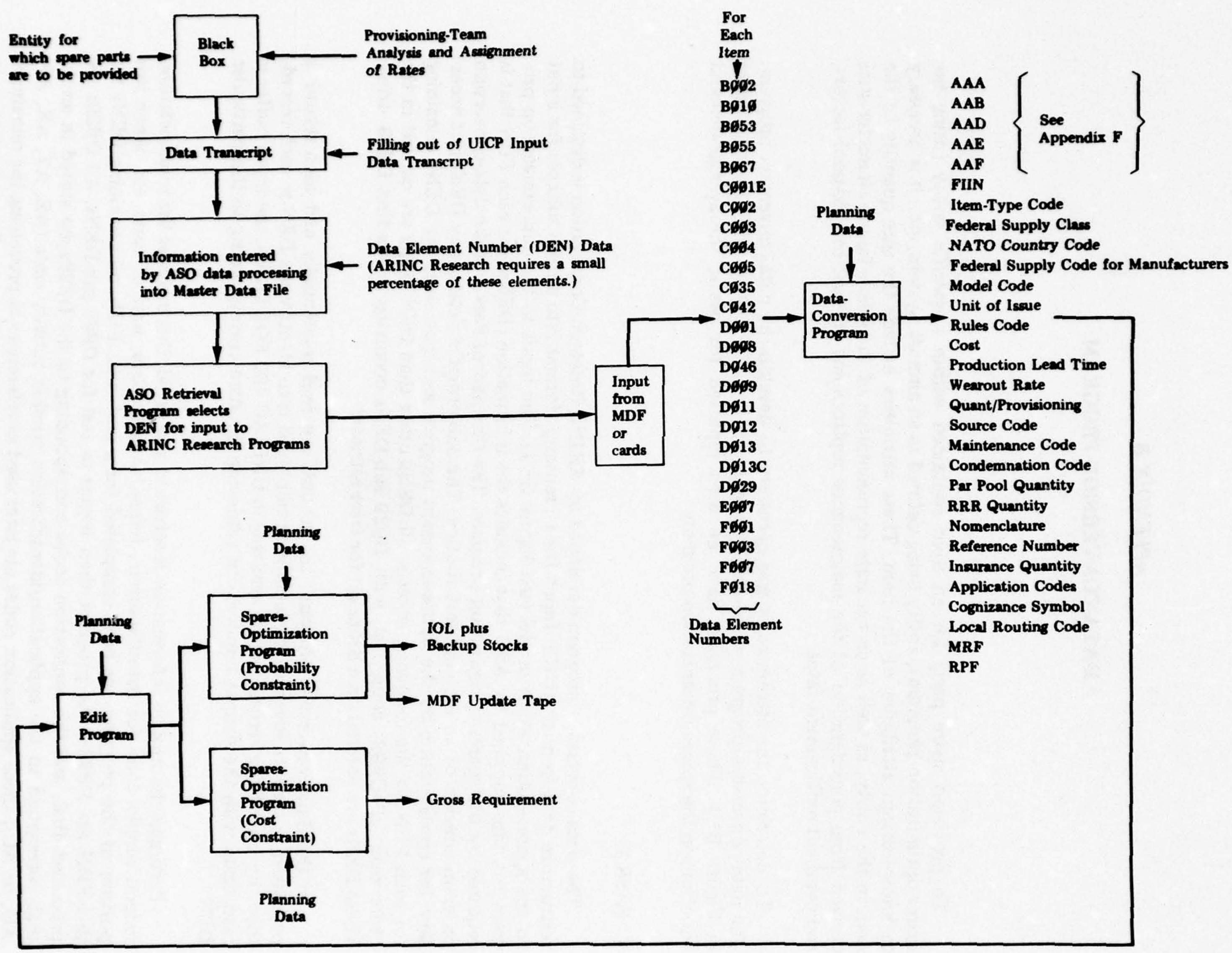


Figure B–1. FUNCTIONAL DIAGRAM (SHOWING DATA FLOW)

is an allowance-list item, is a consumable item, or is a par pool or insurance item. As the remainder of the sub-DENs are processed for a particular FIIN, their data are stored in locations indicated by comparing their application codes with those previously determined as references.

After all records are processed for a particular FIIN, an item-type code is assigned to each FIIN by examination of the SM&R code (DØ13 and DØ13C). These codes indicate that the item is base-repairable, base-consumable, depot-repairable, base/depot-repairable, or depot-consumable. Finally, all the information accumulated for each FIIN is written on tape for input to Pass 2 of the data-conversion program.

2. PASS 2

Pass 2 of the data-conversion program relates all items to higher assemblies and calculates several critical quantities, hereafter called attributes, for each item (See Appendix F). These calculated attributes, along with the identification information of Pass 1, are necessary input data to the ARINC Research Spares Optimization Model, which follows the Data-Conversion Program and the Edit Program (see Appendix E).

Generally, the program separates repairable and consumable items to determine the relationship between highest-level repairables and lower-level repairables and consumables. The consumables are the lowest-level items; thus they are placed in a "save" area to be processed after the higher nesting has been determined.

The highest level of repairables is referred to as "fathers", those having AT in their DØ29 data locations. Each of these is processed immediately as it is read from the input tape (from Pass 1) and transferred to the output tape, since it, as the highest-level repairable, *requires no further processing.* Their model codes, quantities, and calculated number of overhauls are saved to process the next lower level of repairables, the "sons" (those items which have AR in their DØ29 data locations).

The applications for each lower-level repairable are matched against the previous higher level's finished models. The first-level finished models are the "fathers", the second, the "sons", etc. When all of a repairable's application codes have been "satisfied" by relation to a higher assembly, processing for it is complete and it becomes a part of the next higher level of assemblies.

Once a level has been processed, there is no need to keep it, because all references to that level have been satisfied. Hence, the length of the list of finished models may be reduced to a single level, and processing is speeded up.

Upon completion of this process, when all nesting is finished, the model codes, overhauls, and quantities of all processed repairables are returned to core. These will then be matched against the consumables' applications codes. Since the search for a match can be done more quickly by using a *nonsequential search* (binary sectoring), the model codes are sorted from low to high. A search routine can then match up application codes to model codes in a little more than half the time of a sequential search.

The program itself is organized as a set of subroutines called from a main "Driver" program as needed. The subroutines are named INIT, GETFIN, SORT, AECALC, ATCALC, ARCALC, APCALC, and ARERОR.

2.1 Main Driver Routine

The main program, the driver, calls INIT to initialize variables. The driver then loops for each item, calling subroutine GETFIN to read in each item's data and simultaneously checking the item to see if it is a consumable. If it is a consumable, it is saved in a temporary data set, ITAP1R, for further processing. If the item is a repairable, subroutine ATCALC is entered to calculate values for the highest-level repairables (fathers) and for items that have aircraft applications. This loop continues until all of the items have been placed on the final output tape to the Spares Optimization Model (SOM), on JTAP2R (lower-level repairables on which some processing is still required), or on ITAP1R (consumables). The model codes, quantities, and overhauls for all AT items are written on IHOLD and the last-highest-level "hold" tape, JHOLD.

After the FIINs have been processed initially, the repairables tape, JTAP2R, is rewound and reassigned as the input tape, ITAP2R. A "scratch" data set is assigned to JTAP2R for output of incompletely processed repairables. IHOLD is rewound and read, filling JMDLCD, an array in core, which holds the last level of completed repairables. The model codes from the last highest level are sorted. Then, for each repairable remaining to be processed, subroutine ARCALC is entered to relate these items properly to higher-level assemblies that contain them.

When all remaining repairables have been considered, IHOLD is examined to determine if any repairables were processed. If no change was made to IHOLD, but some repairables remain, an error has occurred and subroutine AREROR is called, which attempts to find the error and terminates processing.

When all repairables have been processed, the "hold" tape, JHOLD, is rewound and read to organize all repairables model codes, overhauls, and quantities. These are sorted. The consumables tape, ITAP1R, is rewound. Subroutine APCALC calculates attributes for all consumables using ITAP1R as an input.

The output tape, IFINALTAPE, contains all of the FIINs to be processed by the SOM, and a second tape, NUMBEROFFIINS, containing the number of processed items. These tapes will be the reformatted ASO data input to the ARINC Research spares-optimization Model.

2.2 Individual Subroutines

Subroutine INIT initializes the input/output units, assigning logical units to the symbolic names used in the rest of the program. Planning data are read in from a card: Tau Prime Prime (T2), Systems per Aircraft, and Overhauls per Aircraft. The number of items (FIINS) is read from a tape created in Pass 1. Control returns to the driver program.

Subroutine GETFIN reads the input generated by Pass 1 and finds several printout values — the first occurrence of MRF, RPF, and Percent per Application (PCAP). A list code with 1,6 or A in the fifth character is inserted into the first position of variable JCD; a list code with 2,7 or B goes into the second position of JCD, thus allowing an item to be identified as a Part 1 or a Part 2 IOL item. Control passes back to the driver.

Subroutine SORT sorts, from low to high, an array passed to it as a parameter. A typical example is CALL SORT (ALLMODELS, LENGTH, NOVERHAULS, NSYSTEM). The indexed variable to be sorted is the first parameter, the number of items to be sorted

the second parameter. Then two indexed attributes of the first variable are sorted — the number of overhauls (floating point), and the number (integer) of that item in the system. The sort is performed in a "double bubble" manner, lowest going to the top, highest sinking to the bottom of the sorted array. At completion, control returns to the calling routine.

Subroutine SEARCH looks for the third-fourth parameters (double word) in the first-parameter-named array with length being the second parameter. A typical example is CALL SEARCH (MODELCODES, LENGTH, APPCD(1), APPCD(2), INDEX, ERROR). If successful, the search returns the index of the element in the array and an error code of 0. If unsuccessful, the error code is set to 1. The method of search is binary sectoring, a method superior to sequential searching for a large number of sorted items. Control returns to the calling routine.

Subroutine AECALC calculates attributes for each of the three types of repairables (see Appendix F), using aircraft planning data. These attributes will be summed for each item later. Control returns to the calling routine.

Subroutine ATCALC calculates all attributes for items having AT application codes, highest-level repairables (fathers). If the item is a "father", it is put onto the final output tape, IFINALTAPE, which is used for input to the ARINC Research spares-optimization model (SOM). The model code, calculated overhauls, and quantity of the item are put in a "hold" data set, JHOLD, for later processing of consumables; and in IHOLD, a single-level "hold" data set, used as a reference-data set for lower-level repairables; and if it has aircraft applications, values are calculated for it and then the FIIN is put in a "hold" data set, JTAP2R, for further processing. Control returns to the driver.

Subroutine ARCALC (see Figure B—2) reads from a "hold" data set, ITAP2R (formerly JTAP2R), a partially processed repairable. Application codes not equal to blanks are matched against the last-highest-level processed model codes of JMDLCD, an array in core. If a match of application code (appcode) and model code occurs, the item's attributes are calculated and summed and the particular appcode is set to blanks. When all appcodes are blanks, and processing of the item is complete, its attributes are outputted to the SOM tape (IFINALTAPE), JHOLD, and IHOLD data sets. If there is an unprocessed appcode, the item is placed in JTAP2R for further processing. Control returns to the driver.

Subroutine APCALC calculates the appropriate attributes of a consumable, reading the ITAP2R data set for each item, searching the repairables model codes for an appcode match, and outputting the results to the SOM tape (IFINALTAPE). Control will then return to the driver.

Subroutine AREROR is reached if there is an error in the nesting of repairables. The message "OOPS" appears at the top of a page to signify entrance to ARERÓR. At least one repairable item has an application to a model which was not included in the provisioning. This is an irrecoverable error. Later calculations will become invalid because the repairables below the missing model will be faulty; the consumables below those repairables will be even worse; and the resulting outfitting will be incorrect. The program finds all missing models via application codes and prints them out. If the subroutine is reached, but no message appears, there is a "circular" nesting; e.g., A is contained in B, which is contained in C, which is contained in A, etc. There is no way the program can solve this problem. The program terminates with 9999 displayed in either the missing model code or circular nesting case. Figure B—3 shows the throughput for the data conversion. Program logic is shown in Figures B—4 and B—5, which are followed by a complete program listing.

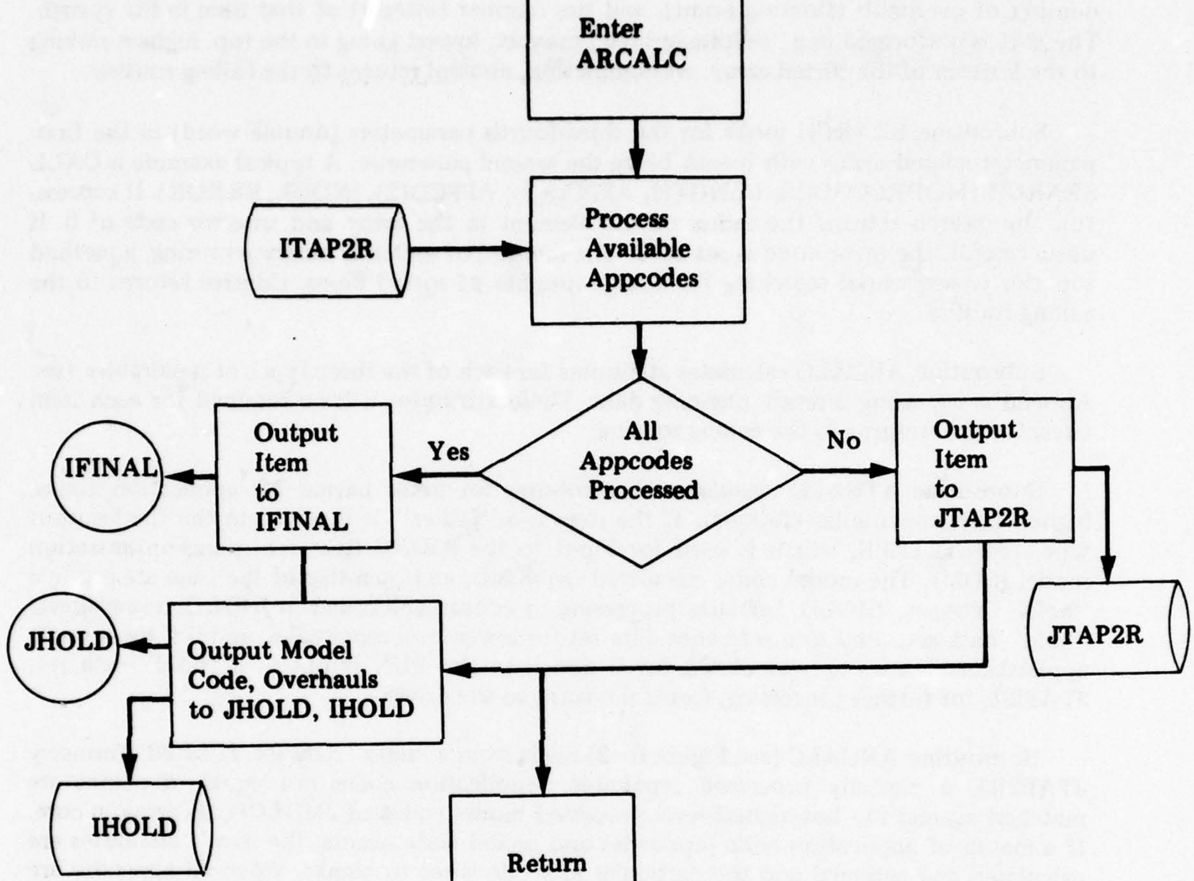


Figure B–2. **SUBROUTINE ARCALC**

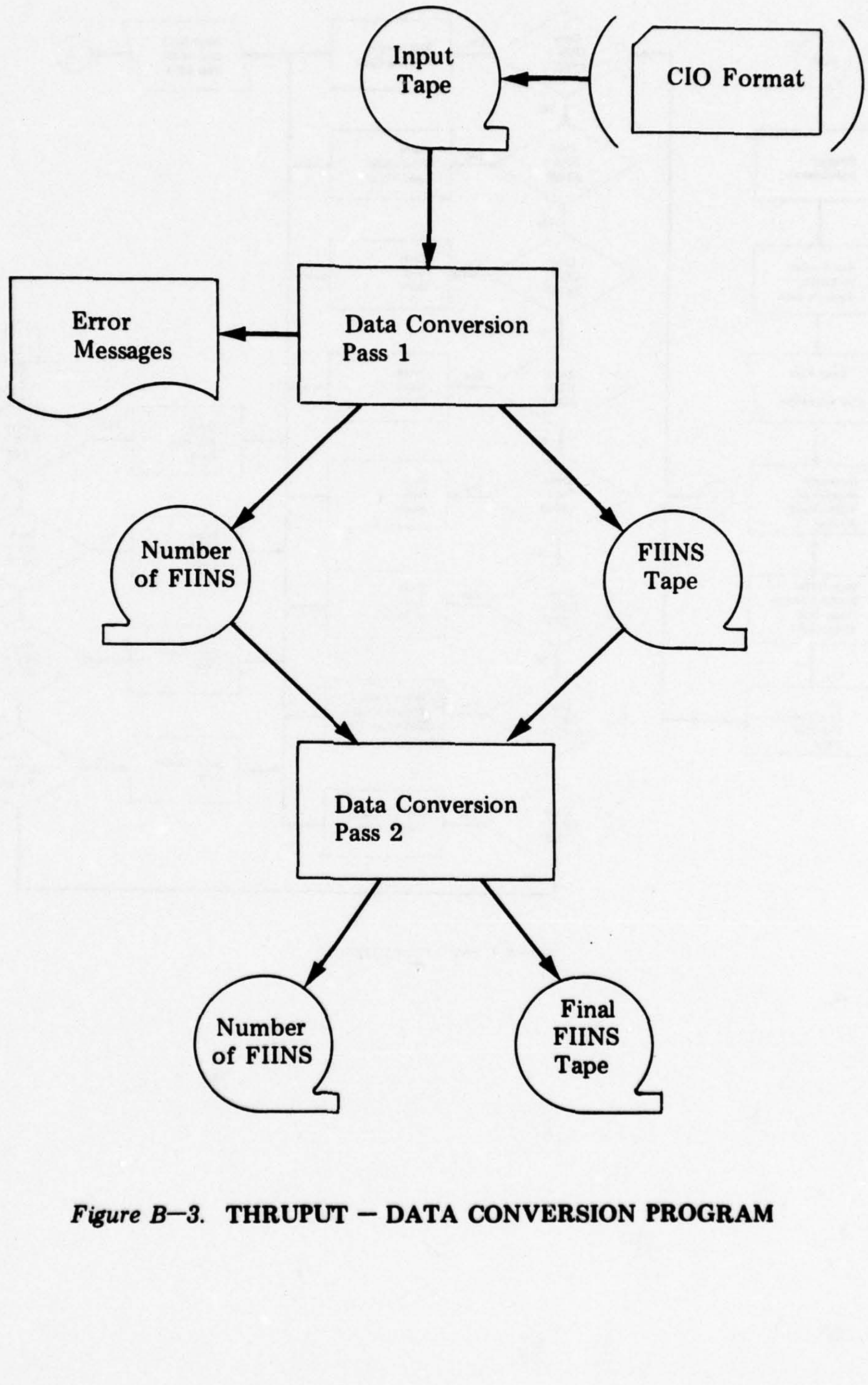


Figure B—3. **THRUPUT — DATA CONVERSION PROGRAM**

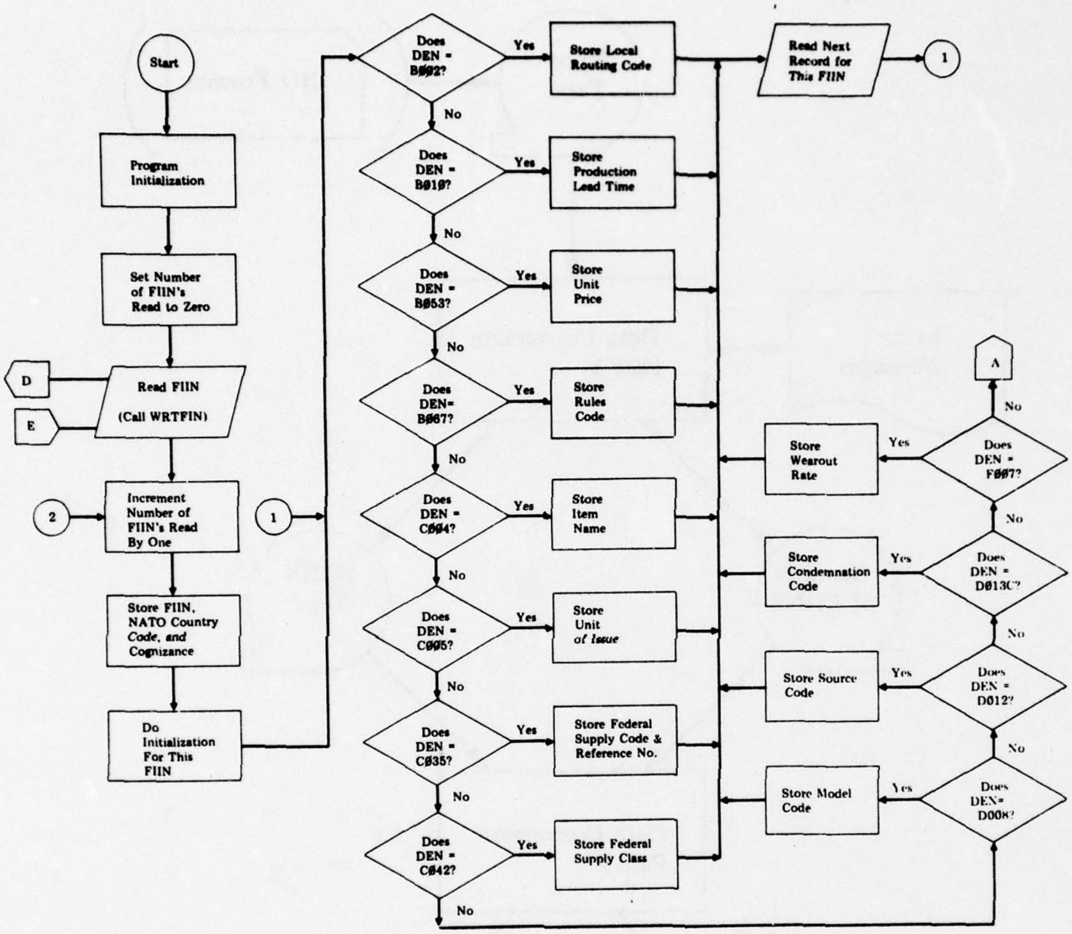


Figure B–4. PASS 1 FLOW CHART

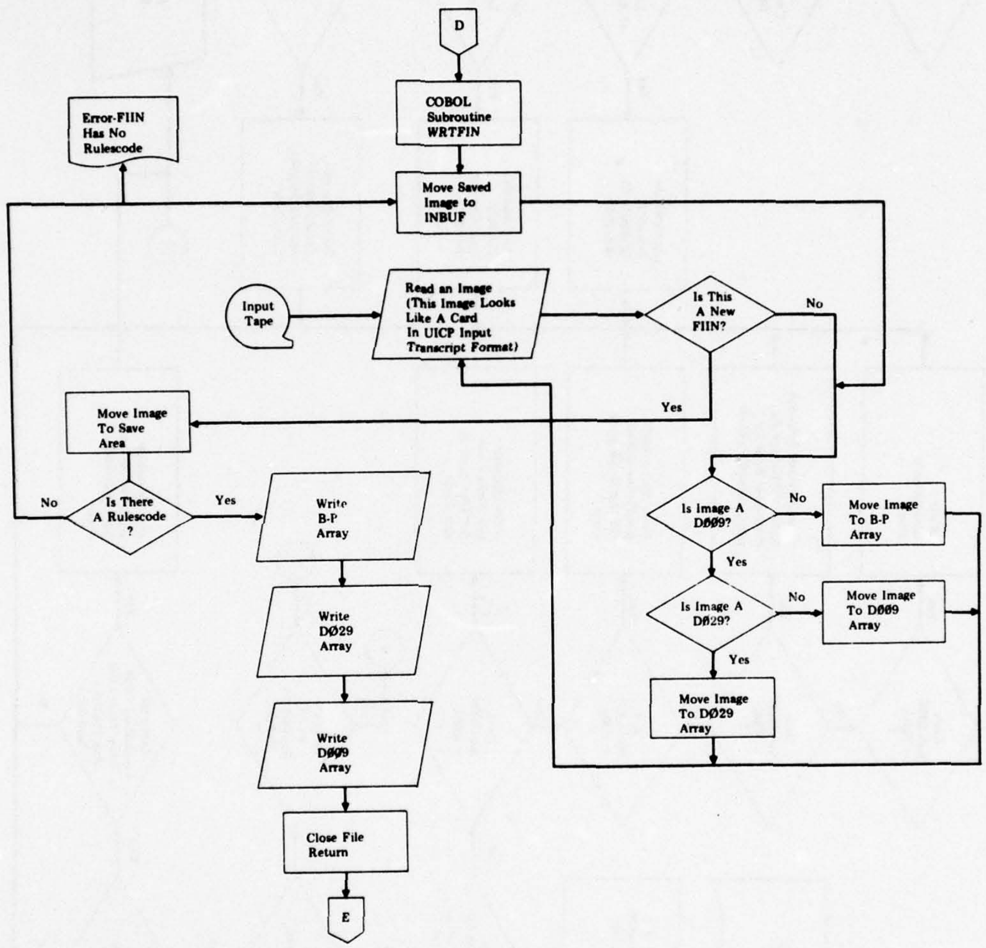


Figure B–4. (continued)

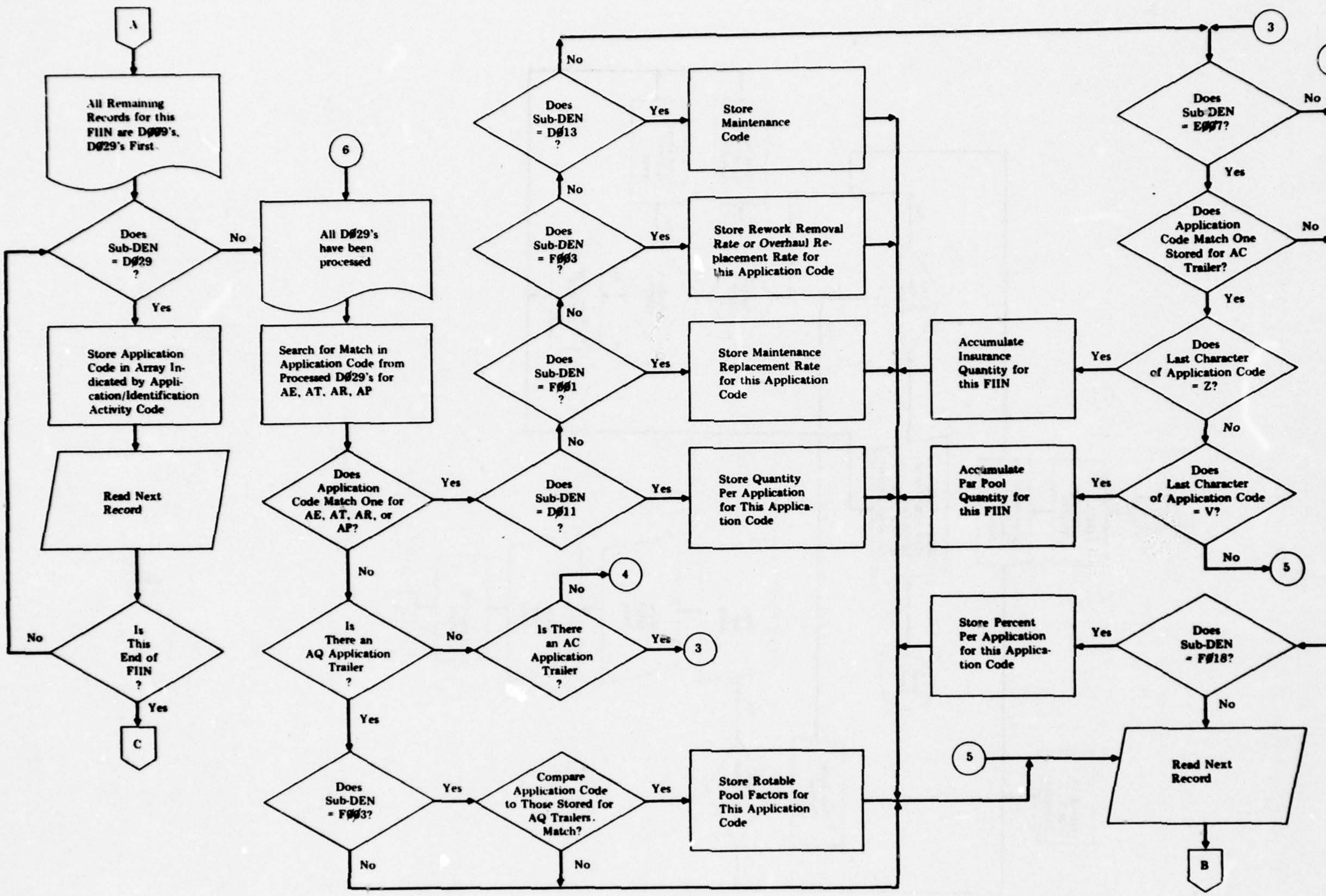


Figure B—4. **(continued)**

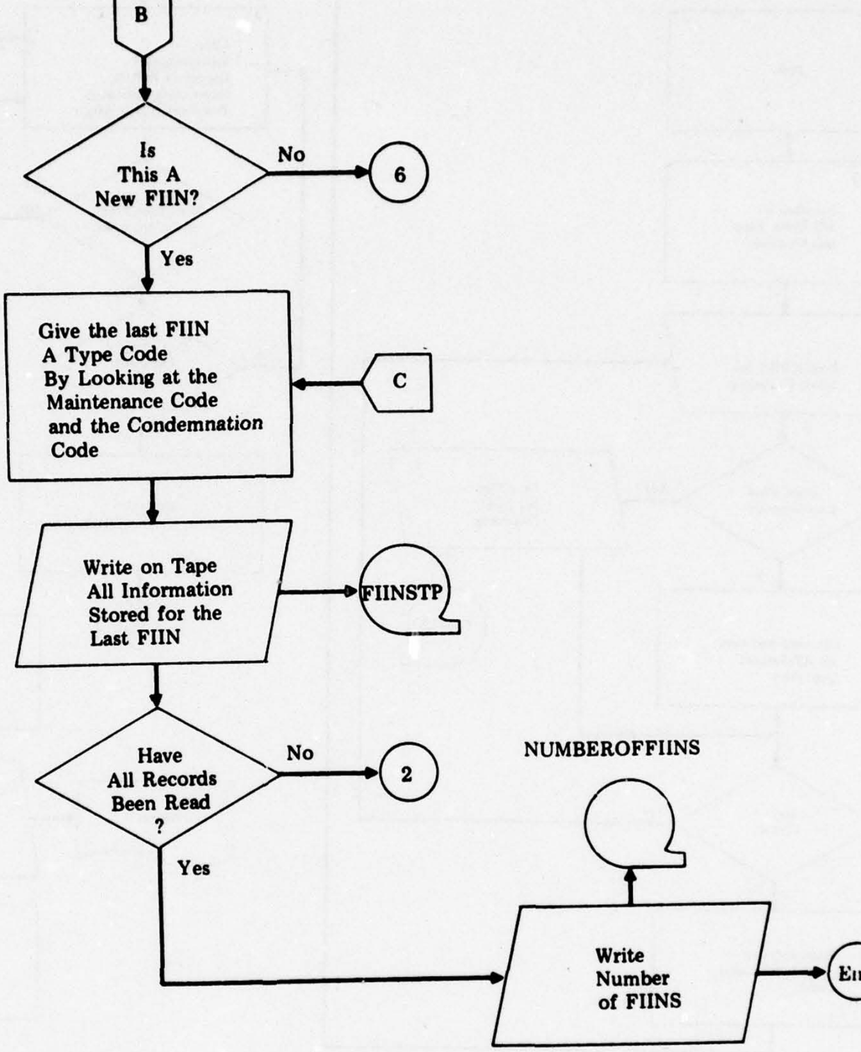


Figure B—4. **(continued)**

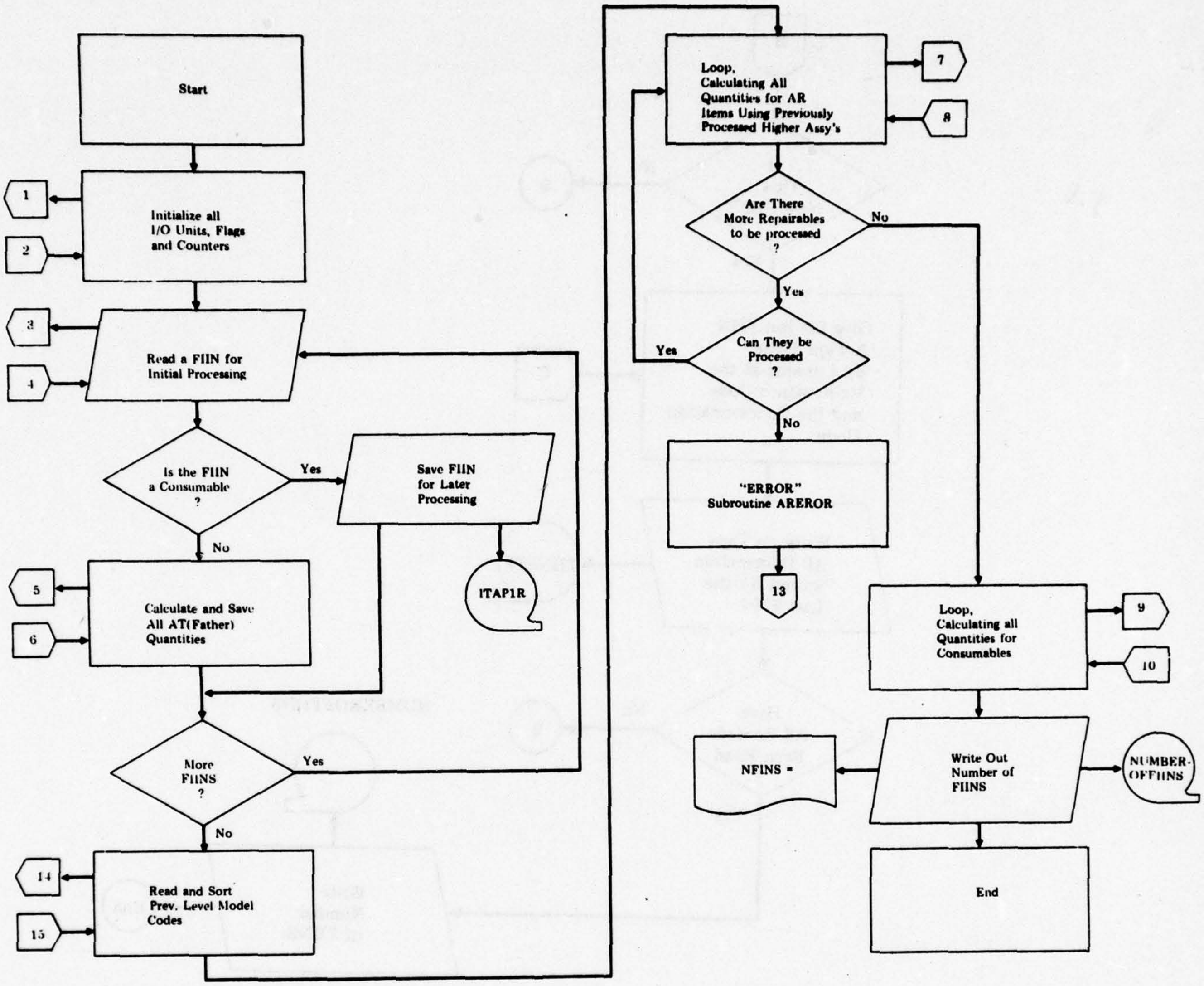


Figure B—5. **PASS 2 FLOW CHART**

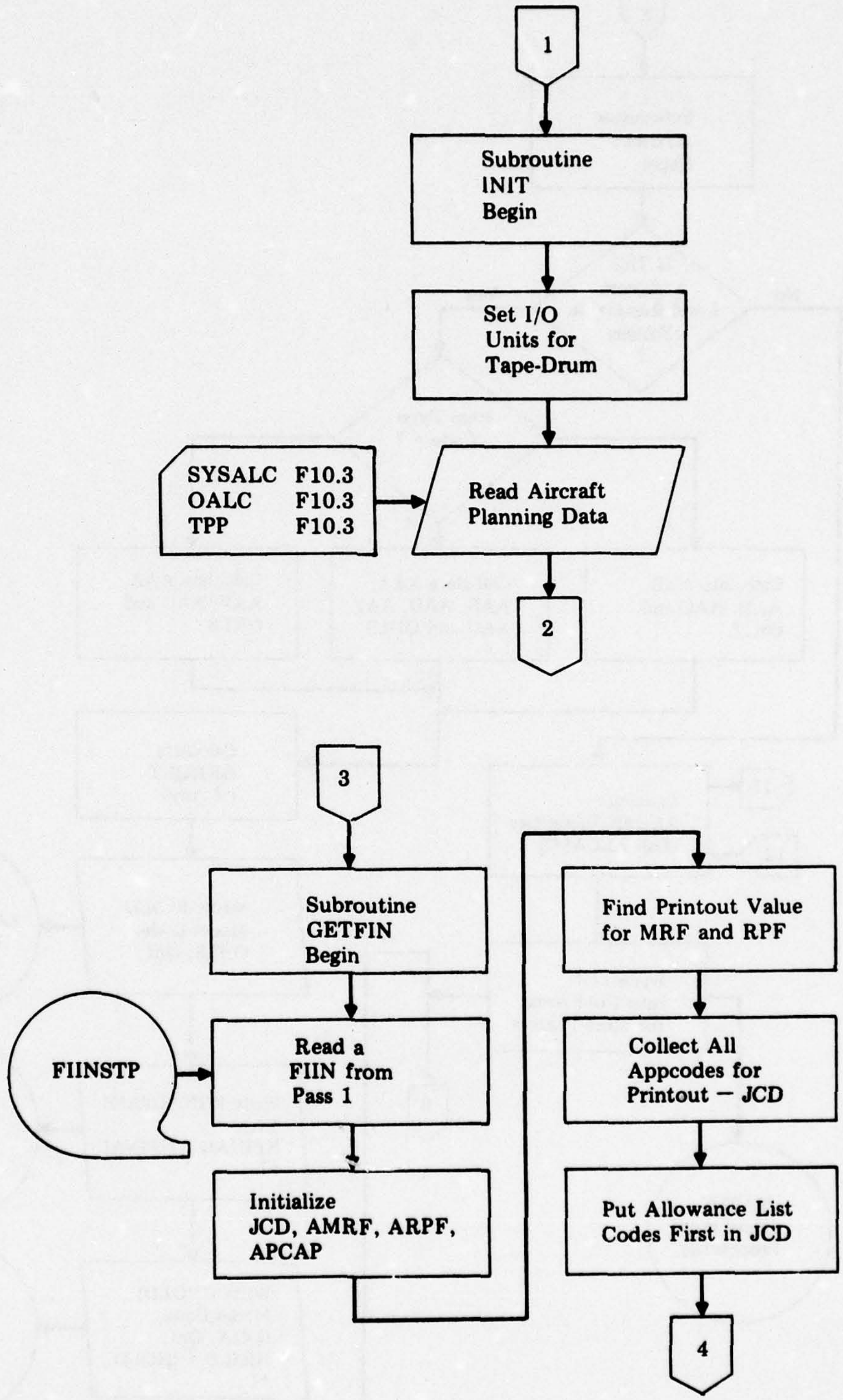


Figure B—5. **(continued)**

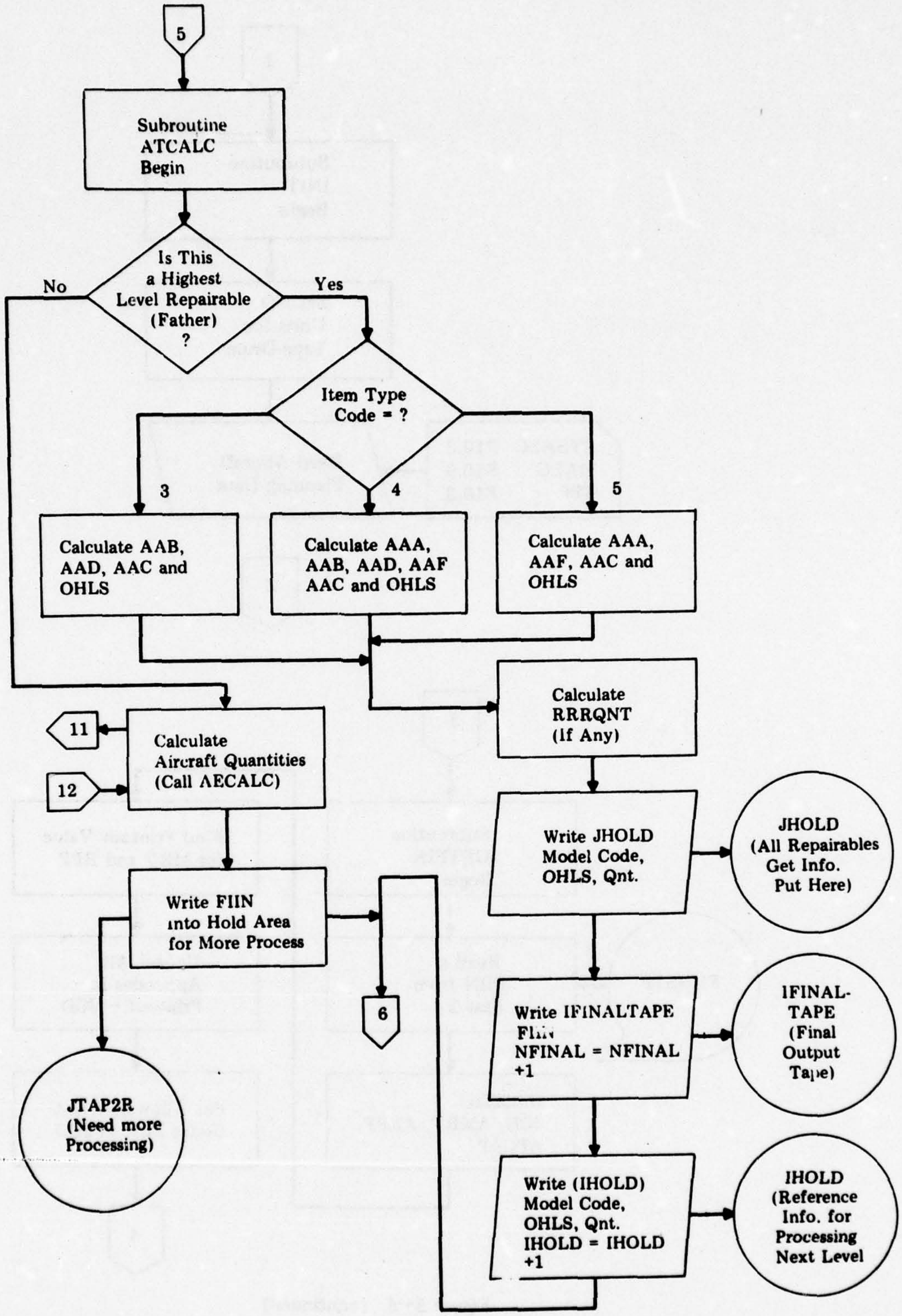


Figure B—5. **(continued)**

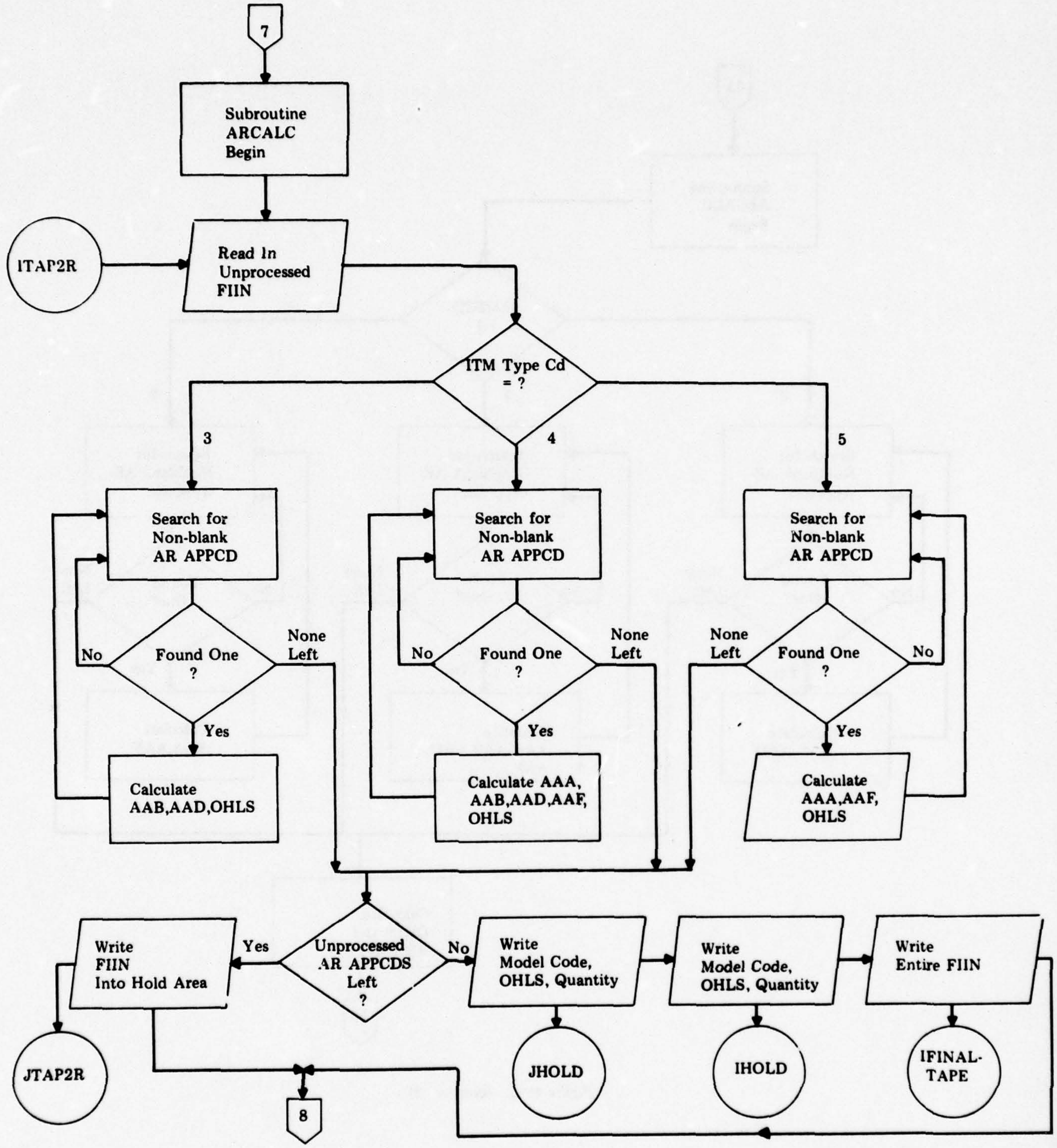


Figure B—5. **(continued)**

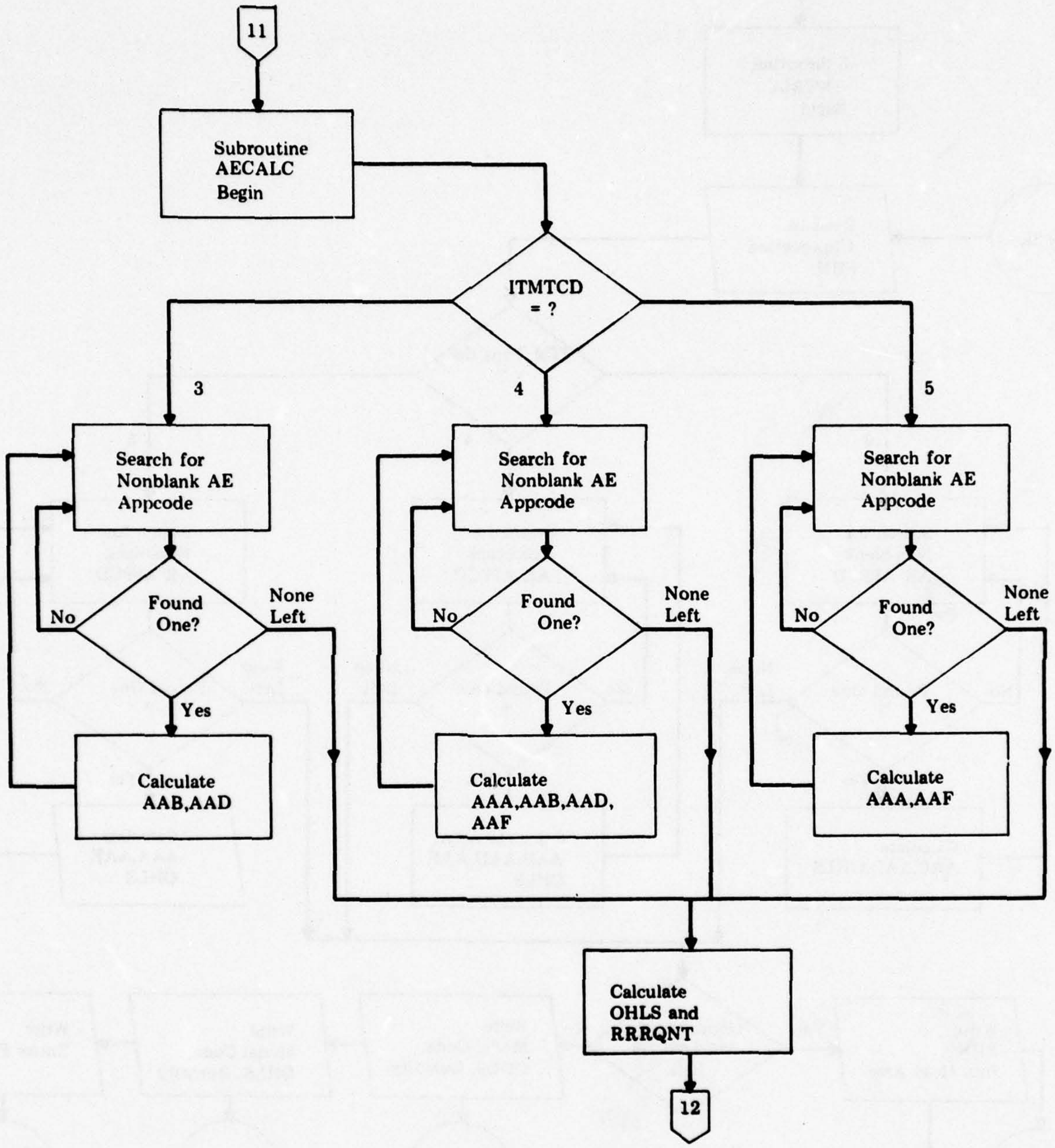


Figure B—5. **(continued)**

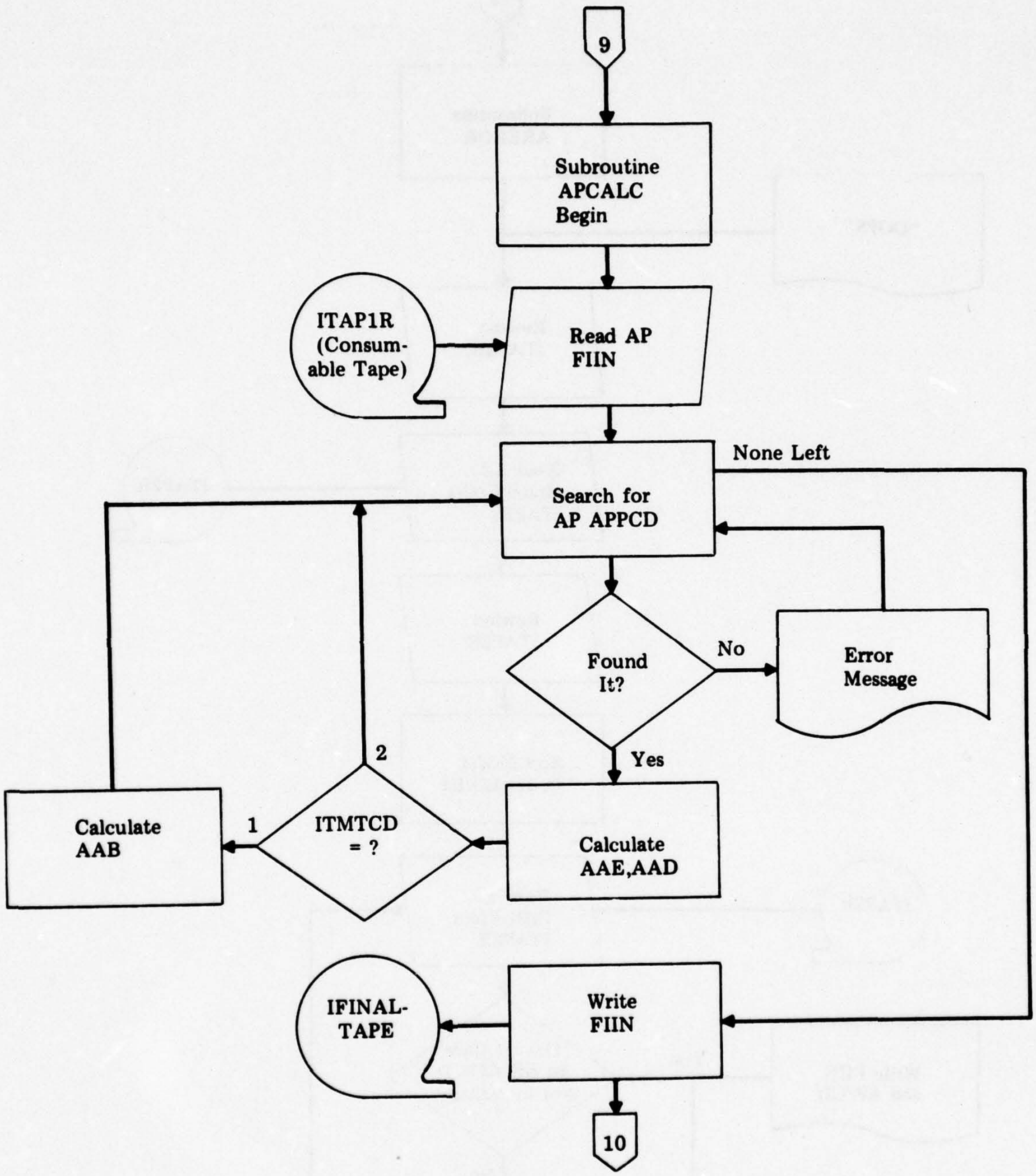


Figure B—5. **(continued)**

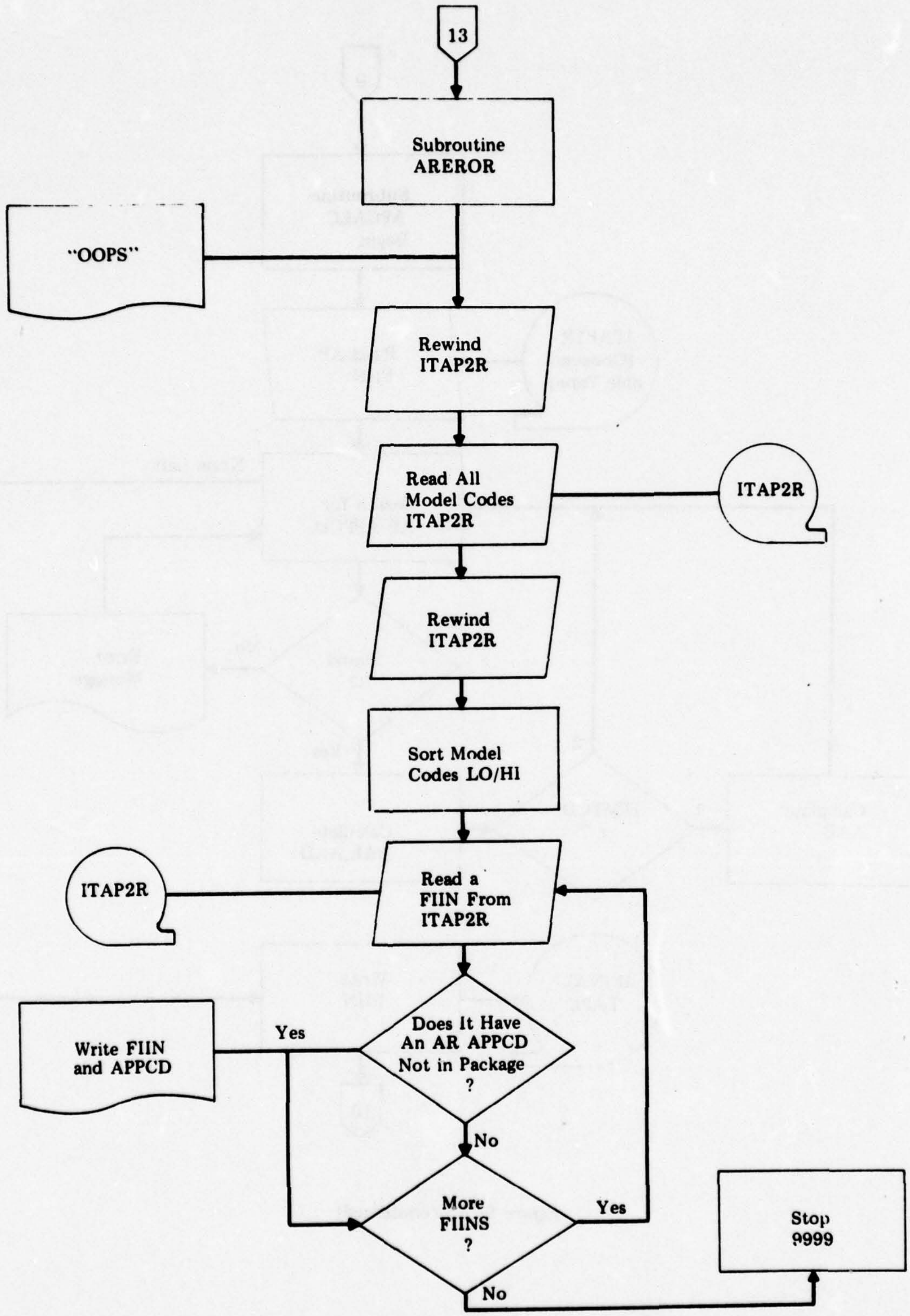
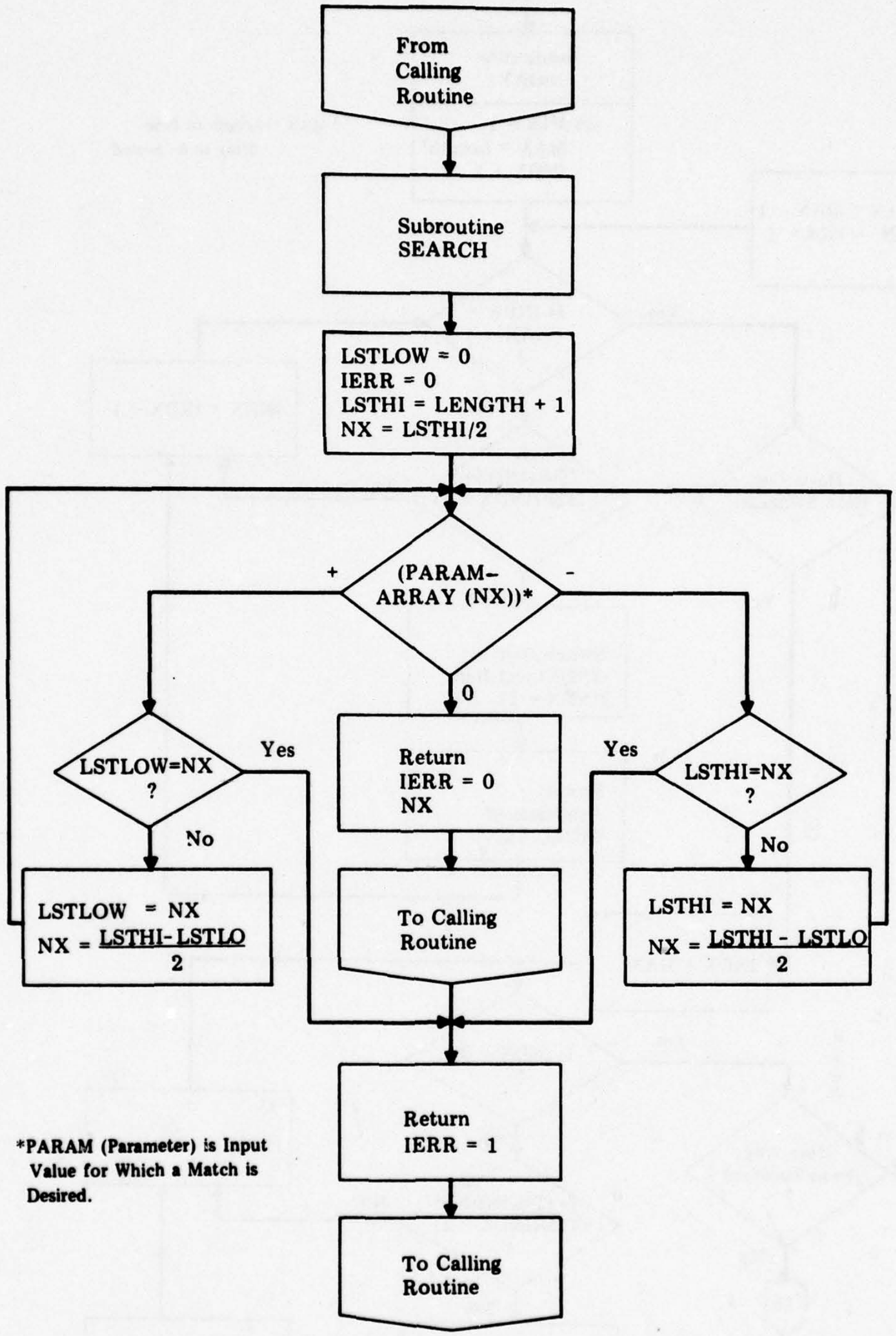


Figure B–5. (continued)



*PARAM (Parameter) is Input Value for Which a Match is Desired.

Figure B–5. **(continued)**

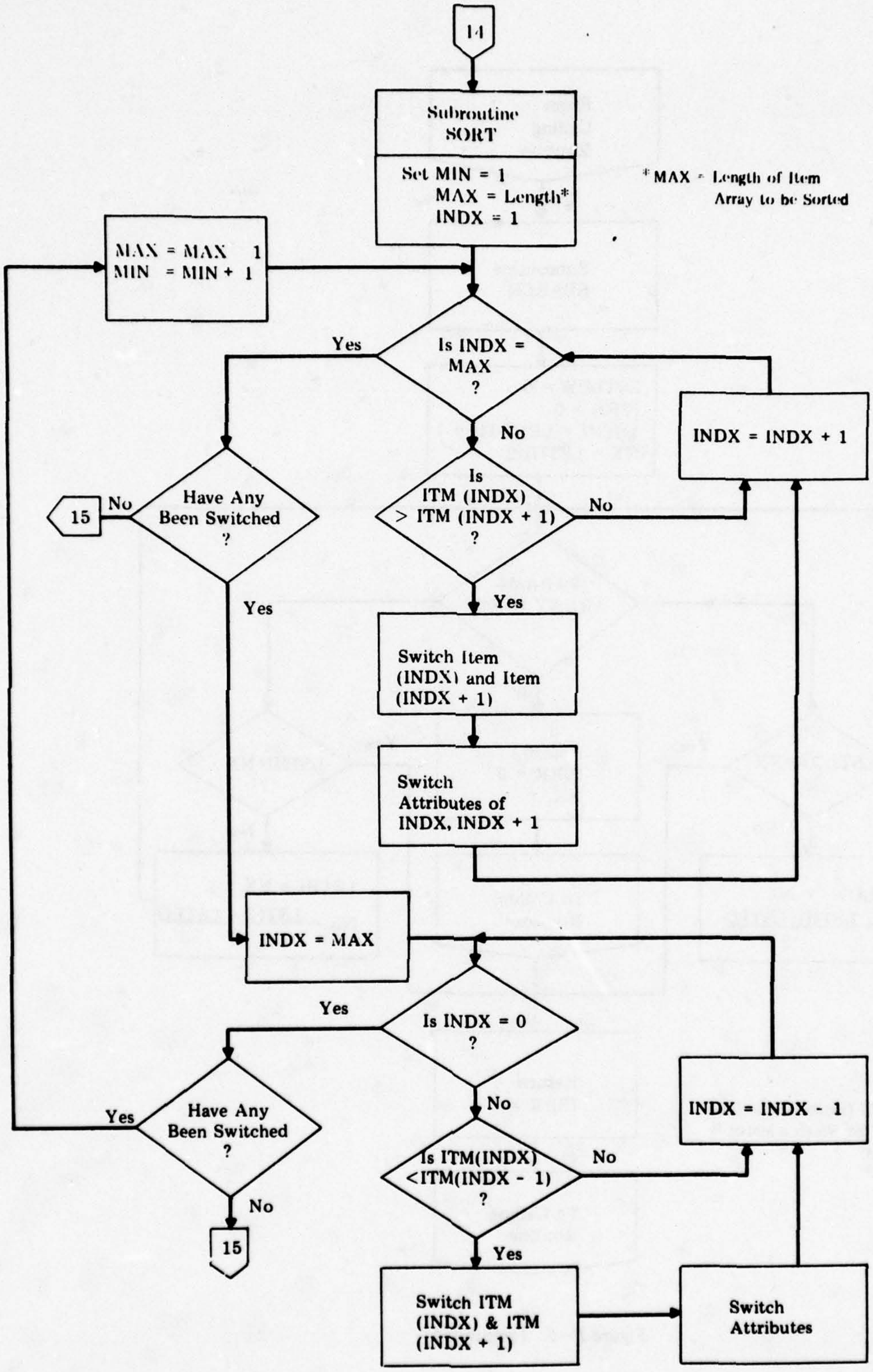


Figure B—5. **(continued)**

```
#FOR Y PROG
         F O R T R A N   I V   C O M P I L A T I O N                       70050 S.
      C   INITIAL FIIN PASS
          REAL MRF
          DIMENSION          RRR(10,4),MRF(10,4),PCAP(10,4),RPF(10)
          INTEGER QNTSYS,COMNCD,B(2,5 ),FSCM(2),SRCCD,ISUMPP,QNTAP(10,4)
          INTEGER APPCD(10,4,2),MNTCD(2    ),APPAC(10,2),APPAQ(10,2),PRPL(10)
          INTEGER FIIN(2),FSC,NCC,MDLCD(2),UNISSU,RULSCD(3),COGSYM
          INTEGER F003,D013,E007,ZDD,DD12,Z,ISAVE(80),B053,C004,C035,D008
          INTEGER D012,DOLLAR,R2B,IR,DOL,D029,B002
          INTEGER    F018,F001A, IAPP(4),D009,AE,AT,AR,AP,AC,AQ,D011,F001,D
          INTEGER F001BK,D013BK,C004BK
          INTEGER B067
          DATA B067/4HB067/
          DATA F001BK/5HF001 /,D013BK/5HD013 /,C004BK/5HC004 /
          DATA LBLK5/5H     /,B002/4HB002/
          DATA D009/4HD009/,D011/4HD011/,F001/4HF001/,F003/4HF003/,AE/2HAE/,
         1 D013/4HD013/,E007/4HE007/,B053/4HB053/,C004/4HC004/,C035/4HC035/,
         2 D008/4HD008/,D012/4HD012/,LBLK/4H    /,D029/4HD029/,F018/4HF018/,
         3 DOL/4H$$$$/,AT/2HAT/,AR/2HAR/,AP/2HAP/,AC/2HAC/,AQ/2HAQ/,D/1HD/,
         4 IR/2H1R/,R2B/2H2R/,Z/1HZ/,NYYY/4HNYYY/



      C
      C   INITIALIZE I/O UNITS
      C
          IOUT=2
          ITAPE=6
          IHOLD=0
          JNP=1
          INFNS=8
          JHOLD=4
          INP=9
          IFNOUT=5
      C   NUMBER OF FIINS(CONSUMABLES+REPAIRABLES) IS NFIN
          NMDLS=1
          NFIN=0
          ISTOP=0
          CALL OPNCDS
        1 CONTINUE
        2 CONTINUE
          REWIND INP
          CALL WRTFIN
```

```
      REWIND INP
C
C   JHOLD IS A SCRATCH AREA ON DRUM 80 CHARACTERS LONG
C
      REWIND JHOLD
      READ(INP,1005) (ISAVE(I),I=1,21)
      READ(IHOLD,1001) COGSYM,NCC,FIIN(1),FIIN(2),ICKY,JDENT
C
C  THE LOOP TERMINATES WITH THE FIRST CARD BEING 4 DOLLAR SIGNS
C
      IF(FIIN(2)-DOL) 1501,999,1501
 1501 CONTINUE
C
C  INITIALIZE ARRAYS AND VARIABLES
C
      DO 1613 I=1,10
      DO 1614 J=1,2
      APPAQ(I,J)=LBLK
      APPAC(I,J)=LBLK
 1614 CONTINUE




 1613 CONTINUE
      DO 2001 I=1,10
      DO 2001 J=1,4
      DO 2001 K=1,2
 2001 APPCD(I,J,K)=LBLK
      DO 8 I=1,10
      NPF(I)=0
      DO 8 J=1,4
      QNTAP(I,J)=0
      RRK(I,J)=0
      MRF(I,J)=0
      PCAP(I,J)=1.0
    8 CONTINUE
      DO 9 I=1,5
      DO 9 J=1,2
    9 B(J,I)=LBLK
      MDLCD(1)=LBLK
      MDLCD(2)=LBLK
      FSCM(1)=LBLK
      FSCM(2)=LBLK
```

```
      MNTCD(1)=LBLK
      MNTCD(2)=LBLK
      SRCCD=LBLK
      CDMNCD=LBLK
      WEROUT=.02
      LCRTCD=LBLK5
      PL=0
      ISUMPP=0
      COST=0
      RULSCD(1)=LBLK
      RULSCD(2)=LBLK
      RULSCD(3)=LBLK
      INSUR=0
      UNISSU=0
      ITMTCD=6
C
C  DON-T WANT TO READ A NEW CARD
C
      GO TO 103
  102 CONTINUE




C
C   READ A NEW CARD FROM THE INPUT FILE
C
      READ(INP,1005)  (ISAVE(I),I=1,21)
      IF(ISAVE(2)-FIIN(1)) 1373,1416,1373
 1416 IF(ISAVE(3)-FIIN(2)) 1373,1417,1373
 1373 WRITE(IOUT,1375) FIIN
 1375 FORMAT(1H0,5HFIIN ,A3,A4,1X,12HHAS NO D009S  )
      GO TO 2
 1417 CONTINUE
      JDENT=ISAVE(6)
C
C  TEST DEN
C
  103 CONTINUE
      IF(JDENT-B002) 105,105,107
  105 READ(IHOLD,1066) LCRTCD
      GO TO 102
  107 CONTINUE
      IF(JDENT-B053) 110,120,130
```

```
110 READ(IHOLD,1004)     PL
    GO TO 102
120 READ(IHOLD,1012)     COST
    GO TO 102
130 IF(JDENT-C004) 140,150,160
140 CONTINUE
    IF(JDENT-B067) 102,142,102
142 CONTINUE
    READ(IHOLD,1014)     RULSCD
    GO TO 102
150 CONTINUE
    READ(IHOLD,152) JDENT
152 FORMAT(13X,A5)
    IF(JDENT-C004BK) 102,155,102
155 CONTINUE
    READ(IHOLD,1013) (B(1,I),I=1,5)
    GO TO 102
160 IF(JDENT-C035) 170,180,182
170 READ(IHOLD,1015)     UNISSU
    GO TO 102




180 READ(IHOLD,1016)     FSCM(1),FSCM(2),(B(2,I),I=1,5 )
    GO TO 102
182 IF(JDENT-D008) 184,186,188
184 READ(IHOLD,1014)     FSC
    GO TO 102
186 READ(IHOLD,1017)     MDLCD(1),MDLCD(2)
    GO TO 102
C
C   IF THE DEN IS LESS THAN D012 IT MUST BE D009, HENCE BRANCH TO NEXT SECT
C
188 IF(JDENT-D012) 200,192,194
192 READ(IHOLD,1015)     SRCCD
    GO TO 102
194 IF(JDENT-D013) 196,196,198
196 READ(IHOLD,1013)     COMNCD
    GO TO 102
198 READ(IHOLD,1003)     WEROUT
    GO TO 102
200 CONTINUE
    IAP1=ISAVE(8)
```

```
      IAP2=ISAVE(9)
      IDENT=ISAVE(11)
      ITEST=ISAVE(13)
 1802 FORMAT(19X,A4,A3,3X,A4,1X,A2)
C
C   ALL THE D009-S FOLLOW, FIRST AMONG THEM ARE D029-S
C
C   FIND ALL D029:S
C
C   IAPP IS THE INDEX ARRAY FOR THE APPCODE ARRAY
C
      IAPPAC=0
      IAPPAQ=0
      DO 1201 I=1,4
 1201 IAPP(I)=0
C
C   LOOP BACK TO HERE TO LOOK FOR NEXT D029
C
 1203 IF(IDENT-D029) 1250,1204,1250
C




C  THERE ARE 6 APPCODES ALLOWED.  CHECK THEM IN TURN
C
 1204 IF(ITEST-AE)1206,1205,1206
 1205 N=1
      GO TO 1212
 1206 IF(ITEST-AT) 1208,1207,1208
 1207 N=2
      GO TO 1212
 1208 IF(ITEST-AR) 1210,1209,1210
 1209 N=3
      GO TO 1212
 1210 IF(ITEST-AP) 1215,1211,1215
 1211 N=4
 1212 IAPP(N)=IAPP(N)+1
      IF(IAPP(N)-11) 1213,1202,1202
 1213 K=IAPP(N)
      APPCD(K,N,1)=IAP1
      APPCD(K,N,2)=IAP2
      GO TO 1202
 1215 IF(ITEST-AQ) 1218,1216,1218
```

```
1216 IAPPAQ=IAPPAQ+1
     IF(IAPPAQ-10) 1217,1217,1202
1217 APPAQ(IAPPAQ,1) =IAP1
     APPAQ(IAPPAQ,2)=IAP2
     GO TO 1202
1218 IF(ITEST-AC) 1202,1219,1202
1219 IAPPAC=IAPPAC+1
     IF(IAPPAC-11) 1220,1202,1202
1220 APPAC(IAPPAC,1)=IAP1
     APPAC(IAPPAC,2)=IAP2
1202 CONTINUE
C
C   GET A NEW CARD
C
     READ(INP,1005)  (ISAVE(I),I=1,21)
C
C    CHECK IF IT IS THE SAME FIIN
C
C   CHECK IF END OF RECORD
     IF(ISAVE(3)-DOL) 1247,600,1247




 1247 IAP1=ISAVE(8)
      IAP2=ISAVE(9)
      IDENT=ISAVE(11)
      ITEST=ISAVE(13)
      GO TO 1203
C
C   IF NOT SAME FIIN NO NEED TO PROCESS NON-D029S BECAUSE THERE NONE
C
C
C   PROCESS NON-D029 CARDS WITH DEN OF D009
C
 1250 CONTINUE
      IDENT=ISAVE(6)
      IAP1=ISAVE(8)
      IAP2=ISAVE(9)
      JDENT=ISAVE(11)
C
C   CHECK FOR CHANGE OF FIIN
C
C
```

```
C    SEARCH FOR APPCODE MATCH WITH ALREADY PROCESSED D029
C
  402 DO 410 IAP=1,4
      K=IAPP(IAP)
      IF(K) 410,410,403
  403 DO 409 J=1,K
      IF(APPCD(J,IAP,1)-IAP1) 409,404,409
  404 IF(APPCD(J,IAP,2)-IAP2) 409,540,409
  409 CONTINUE
  410 CONTINUE
  420 IF(IAPPAQ) 430,430,422
  422 CONTINUE
      IF(JDENT-F003) 430,511,430
  511 DO 514 I=1,IAPPAQ
      IF(IAP1  -APPAQ(I,1)) 514,512,514
  512 IF(IAP2  -APPAQ(I,2))514,513,514
  513 READ(IHOLD,1094) RPF(I)
 1094 FORMAT(34X,F3.2)
      GO TO 580
  514 CONTINUE
      GO TO 580
  540 IF(JDENT-D011) 545,542,545
  542 READ(IHOLD,1065)    QNTAP(J,IAP)
      GO TO 580
  545 IF(JDENT-F001) 550,547,550
  547 READ(IHOLD,546) JDENT
  546 FORMAT(29X,A5)
      IF(JDENT-F001BK) 580,3545,580
 3545 CONTINUE
      READ(IHOLD,1030) MRF(J,IAP)
      GO TO 580
  550 IF(JDENT-F003) 555,552,555
  552 READ(IHOLD,1034)    RRR(J,IAP)
      GO TO 580
  555 IF(JDENT-D013) 557,556,557
  556 READ(IHOLD,546) JDENT
      IF(JDENT-D013BK) 580,3556,580
 3556 CONTINUE
      READ(IHOLD,1031) MNTCD(1),MNTCD(2)
      GO TO 580
```

```
430 IF(IAPPAC) 570,570,557
557 IF(JDENT-E007) 570,558,570
558 DO 565 I=1,IAPPAC
    IF(IAP1  -APPAC(I,1))565,559,565
559 IF(IAP2  -APPAC(I,2))565,560,565
560 READ(IHOLD,1033) KTEST,IAA,IAC
    IF(KTEST-Z) 563,561,563
561 INSUR=INSUR+IAA+IAC
    GO TO 580
563 ISUMPP=ISUMPP+IAA+IAC
    GO TO 580
565 CONTINUE
570 IF(JDENT-F018) 580,571,580
571 READ(IHOLD,1034) PCAP(J,IAP)
580 CONTINUE
C
C   GET NEW CARD FOR PROCESSING
C
    READ(INP,1005)  (ISAVE(I),I=1,21)
C

C   CHECK FOR END OF RECORD
C
    IF(ISAVE(3)-DOL) 1250,600,1250
600 CONTINUE
C
C   FIND THE MAINTENANCE ITEM TYPE CODE FOR THIS FIIN
C
    DD12=2
    IF(MNTCD(1)-LBLK) 627,652,627
627 IF(MNTCD(2)-LBLK) 628,652,628
628 CONTINUE
    IF(CDMNCD-LBLK) 630,652,630
630 CONTINUE
    DD12=1
    ITEST1=MNTCD(2)
    IF(ITEST1-Z) 632,631,632
631 ZDD=1
    GO TO 655
632 IF(ITEST1-D) 634,633,634
633 ZDD=2
```

```
      GO TO 640
  634 ZDD=3
  640 IF(CDMNCD-D) 642,645,642
  642 DD12=2
  645 GO TO (651,652,654),ZDD
  651 ITMTCD=DD12
      GO TO 660
  652 ITMTCD=3*DD12
      GO TO 660
  654 ITMTCD=DD12+3        655 ITEST2=MNTCD(1)
             GO TO 660
  660 CONTINUE                 IF(ITEST2-D) 642,645,642
C     CHECK TO SEE IF REPAIRABLE HAS MODEL CODE
 1300 IF(MDLCD(1)-LBLK) 1305,1301,1305
 1301 IF(MDLCD(2)-LBLK) 1305,1302,1305
C     NO MODEL CODE-IS THIS A CONSUMABLE
 1302 IF(RULSCD(3)-NYYY) 1305,1305,1303
 1303 WRITE(IOUT,1304) FIIN(1),FIIN(2)
 1304 FORMAT(1H0,10X,4HFIIN,2X,A4,A3,45HHAS NO MODEL CODE AND HAS NOT BE
     1EN PROCESSED         )
      ISTOP=ISTOP+1
      GO TO 2
 1305 CONTINUE
      IF (ITMTCD-6) 1307,1306,1306
 1306 CONTINUE
      WRITE(IOUT,1309) FIIN(1),FIIN(2)
 1309 FORMAT(1H0,10X,5HFIIN A3,A4,26H IS AN UNKNOWN ITEM TYPE         )
      GO TO 2
 1307 CONTINUE
C
C     OUTPUT ENTIRE FIIN RECORD, UNFORMATTED
C
      WRITE(IFNOUT) FIIN,ITMTCD,FSC,NCC,FSCM,MDLCD,UNISSU,RULSCD,COGSYM,
     1 COST,PL,WEROUT,APPCD,MRF,RRR,PCAP,MNTCD,QNTAP,RPF,APPAC,APPAQ,
     2        COMNCD,B,SRCCD,ISUMPP,INSUR,LCRTCD
      NFIN=NFIN+1
C
C
C     BRANCH BACK TO MAIN LOOP
C
      GO TO 1
```

```
  999 CONTINUE
      WRITE(IOUT,1081) NFIN,NMDLS
      ENDFILE IFNOUT
      WRITE(INFNS,1311) NFIN,NMDLS
 1311 FORMAT(2I10)
      REWIND INFNS
      IF(ISTOP) 1274,1274,1575
 1575 CONTINUE
      WRITE(IOUT,1293) ISTOP
 1293 FORMAT(1H1,18HJOB ABORTED DUE TO    ,I5,20H MISSING MODEL CODES    )
 1274 CONTINUE
 1001 FORMAT(2A2,A3,A4,A2,A4)
 1002 FORMAT(13X,A4)
 1003 FORMAT(19X,F3.2)
 1004 FORMAT([illegible]X,F3.1)
 1005 FORMAT(A4,A3,A4,2A1,A4,A2,A4,2A3,A4,A1,A2,11A4)
 1012 FORMAT(19X,F9.2)
 1013 FORMAT(19X, 5A4)
 1014 FORMAT(19X,A4,A2,A4)
 1015 FORMAT(19X,A2)
 1016 FORMAT(19X,A4,A1, 5A4)
 1017 FORMAT(19X,A4,A3)
 1021 FORMAT(1X,A1)
 1030 FORMAT(34X,F6.3)
 1031 FORMAT(34X,2A1)
 1033 FORMAT(25X,A1,14X,I3,6X,I3)
 1034 FORMAT(34X,F3.2)
 1040 FORMAT(19X,A4,A3,3X,A4,1X,A2)
 1041 FORMAT(11A4,6A2,A1,A4,A1,2A2, 2A4,A1,A4)
 1046 FORMAT(A5,A4)
 1050 FORMAT(1H1,(5X,A4,A3,4X,I5,3X,6F10.2,2X,A4/))
 1052 FORMAT(1H0,3X,A4,A3,10X,3(F15.5,5X),A2)
 1060 FORMAT(2X,(10X,2A4,2X,I6,10X,4F15.5/))
 1065 FORMAT(34X,I6)
 1066 FORMAT(19X,A5)
 1081 FORMAT(1H1,10X,5HNFIN=,I10,3X,7HMODELS=,I10)
      STOP
      END
#END
```

```
#COB BLVY PROG1

     UNIVAC 490/491/492/494  COBOL COMPILATION    DATE-70050     TIME-00:00              VERGC
          000010 IDENTIFICATION DIVISION.
                 PROGRAM-ID. ARINC DATA CONVERSION PROGRAM.
          000030 ENVIRONMENT DIVISION.
          000040 CONFIGURATION SECTION.
          000050 SOURCE-COMPUTER. UNIVAC-492.
          000060 OBJECT-COMPUTER. UNIVAC-492.
          000065 INPUT-OUTPUT SECTION.
          000070 FILE-CONTROL.
          000075       SELECT INCARDS ASSIGN TO F TAPE.
                       SELECT OUT-DATA ASSIGN TO I DRUM.
                 DATA DIVISION.
                 FILE SECTION.
                 FD  INCARDS
                     LABEL RECORDS ARE OMITTED
                     DATA RECORD IS CDS.
                 01  CDS.
                     03 IN-REC.
                         05 FILLER          PICTURE X(4).
                         05 REC-ID          PICTURE X(7).
                         05 FILLER          PICTURE X(2).
                         05 MAJ-ID          PICTURE X(5).
                         05 FILLER          PICTURE X(11).
                         05 MIN-ID          PICTURE X(4).
                         05 FILLER          PICTURE X(47).
                     03 FILLER              PICTURE X(55).
                 FD  OUT-DATA
                     LABEL RECORDS ARE OMITTED
                     ACCESS MODE IS SEQUENTIAL
                     DATA RECORD IS OUT-REC.
                 01  OUT-REC.
                     03 TBL-LINE            PICTURE X(80)  VALUE SPACES.
                     03 FILLER              PICTURE X(55).
                 WORKING-STORAGE SECTION.
                     77 REC-ID-HLD          PICTURE X(7)   VALUE SPACES.
                     77 IDY                 PICTURE 999    VALUE ZEROS.

                     77 IDX                 PICTURE 999    VALUE ZEROS.
                     77 IDZ                 PICTURE 999    VALUE ZEROS.
                 01  DOLLAR-LINE.
                     03 FILLER              PICTURE X(10) VALUE '$$$$$$$$$$'.
                     03 FILLER              PICTURE X(10) VALUE '$$$$$$$$$$'.
                     03 FILLER              PICTURE X(10) VALUE '$$$$$$$$$$'.
                     03 FILLER              PICTURE X(10) VALUE '$$$$$$$$$$'.
                     03 FILLER              PICTURE X(10) VALUE '$$$$$$$$$$'.
                     03 FILLER              PICTURE X(10) VALUE '$$$$$$$$$$'.
                     03 FILLER              PICTURE X(10) VALUE '$$$$$$$$$$'.
                     03 FILLER              PICTURE X(10) VALUE '$$$$$$$$$$'.
                     77 CHK-SWT             PICTURE 9      VALUE ZEROS.
                     77 DOLLAR-SWT          PICTURE 9      VALUE ZEROS.
                     77 SPACER              PICTURE X(7)   VALUE SPACES.
                 01  MAJ-ID-TBL.
                     03 FILLER              PICTURE X(5)   VALUE 'D009 '.
                     03 FILLER              PICTURE X(5)   VALUE 'B067 '.
                     03 FILLER              PICTURE X(5)   VALUE 'C042 '.
                     03 FILLER              PICTURE X(5)   VALUE 'C005 '.
                     03 FILLER              PICTURE X(5)   VALUE 'B053 '.
                     03 FILLER              PICTURE X(5)   VALUE 'B055 '.
                     03 FILLER              PICTURE X(5)   VALUE 'B002 '.
                     03 FILLER              PICTURE X(5)   VALUE 'D012 '.
                     03 FILLER              PICTURE X(5)   VALUE 'C004 '.
                     03 FILLER              PICTURE X(5)   VALUE 'B010 '.
                     03 FILLER              PICTURE X(5)   VALUE 'D013C'.
                     03 FILLER              PICTURE X(5)   VALUE 'C035 '.
                     03 FILLER              PICTURE X(5)   VALUE 'D008 '.
                     03 FILLER              PICTURE X(5)   VALUE 'F007 '.
                 01  DUMMY REDEFINES MAJ-ID-TBL.
                     03 MAJ                 PICTURE X(5)   OCCURS 14 TIMES.
                     77 INDX                PICTURE 99     VALUE ZEROS.
                     77 RULESCODE           PICTURE 9      VALUE ZEROS.
                     77 EROR     PICTURE X(18) VALUE ' HAS NO RULESCODE.'.
                 01  B-P-TBL.
                     03  B-P OCCURS 50 TIMES.
                         05 FILLER          PICTURE X(4).
                         05 B-P-ID          PICTURE X(7).
                         05 FILLER          PICTURE X(69).
                 01  D009-TBL.
```

```
           03 D009-REC OCCURS 150 TIMES.
              05 FILLER              PICTURE X(4).
              05 D009-REC-ID         PICTURE X(7).
              05 FILLER              PICTURE X(2).
              05 D009-MAJ-ID         PICTURE X(5).
              05 FILLER              PICTURE X(11).
              05 D009-MIN-ID         PICTURE X(4).
              05 FILLER              PICTURE X(47).
       01 HOLD-REC.
           03  FILLER                PICTURE X(4)   VALUE SPACES.
           03  HOLD-ID               PICTURE X(7)   VALUE SPACES.
           03  FILLER                PICTURE X(69)  VALUE SPACES.
       01  D029-TBL.
           03 D029-REC OCCURS 50 TIMES.
              05 FILLER              PICTURE X(4).
              05 D029-REC-ID         PICTURE X(7).
              05 FILLER              PICTURE X(2).
              05 D029-MAJ-ID         PICTURE X(5).
              05 FILLER              PICTURE X(11).
              05 D029-MIN-ID         PICTURE X(4).
              05 FILLER              PICTURE X(47).
       PROCEDURE DIVISION.
       WRTFIN.
           OPEN OUTPUT OUT-DATA.
           ENTER FIXTBL, WITH OUT-DATA.
           IF DOLLAR-SWT EQUAL TO 1
              CLOSE INCARDS
              GO TO RTN.
       NXT-FIN.
           MOVE ZEROS TO RULESCODE.
           MOVE ZEROS TO INDX.
           MOVE ZEROS TO IDY.
           MOVE ZEROS TO IDZ.
           MOVE HOLD-REC TO IN-REC.
           MOVE HOLD-ID TO REC-ID-HLD.
           GO TO ANAL.
       BEGIN.
           READ INCARDS AT END GO TO EOF-DTA.
           IF REC-ID NOT EQUAL TO REC-ID-HLD
              GO TO SAVE-REC.
```

```
       ANAL.
           MOVE ZEROS TO CHK-SWT.
           PERFORM CHECK-MAJOR-ID VARYING IDX FROM 1 BY 1 UNTIL
              IDX EQUAL TO 15.
           IF CHK-SWT EQUAL TO ZEROS
              GO TO BEGIN.
           IF MAJ-ID EQUAL TO 'D009 ' AND MIN-ID EQUAL TO 'D029'
              ADD 1 TO IDZ
              MOVE IN-REC TO D029-REC (IDZ)
              GO TO BEGIN.
           IF MAJ-ID EQUAL TO 'D009 '
              ADD 1 TO IDY
              MOVE IN-REC TO D009-REC (IDY)
              GO TO BEGIN.
           ADD 1 TO INDX.
           MOVE IN-REC TO B-P (INDX).
           IF MAJ-ID EQUAL TO MAJ (2)
              MOVE 1 TO RULESCODE.
           GO TO BEGIN.
       CHECK-MAJOR-ID.
           IF MAJ-ID EQUAL TO MAJ (IDX)
              MOVE 1 TO CHK-SWT.
       SAVE-REC.
           MOVE IN-REC TO HOLD-REC.
           IF RULESCODE NOT EQUAL TO ZEROS
              GO TO NXT-B-P.
           IF INDX NOT EQUAL TO ZEROS
              DISPLAY B-P-ID (1), EROR
              GO TO NXT-FIN.
           IF IDZ NOT EQUAL TO ZEROS
              DISPLAY D029-REC-ID (1), EROR
              GO TO NXT-FIN.
           IF IDY NOT EQUAL TO ZEROS
              DISPLAY D009-REC-ID (1), EROR
              GO TO NXT-FIN.
       NXT-B-P.
           IF INDX EQUAL TO ZERO
              GO TO D029-OUT.
           MOVE B-P (INDX) TO TBL-LINE.
           WRITE OUT-REC.
```

```
                   SUBTRACT 1 FROM INDX.
                   GO TO NXT-B-P.
               D029-OUT.
                   IF IDZ EQUAL TO ZEROS
                     GO TO D009-OUT.
                   MOVE D029-REC (IDZ) TO TBL-LINE.
                   WRITE OUT-REC.
                   SUBTRACT 1 FROM IDZ.
                   GO TO D029-OUT.
               D009-OUT.
                   IF IDY EQUAL TO ZEROS
                     GO TO RTN.
                   MOVE D009-REC (IDY) TO TBL-LINE.
                   WRITE OUT-REC.
                   SUBTRACT 1 FROM IDY.
                   GO TO D009-OUT.
               RTN.
                   MOVE DOLLAR-LINE TO TBL-LINE.
                   WRITE OUT-REC.
                   WRITE OUT-REC.
                   WRITE OUT-REC.
                   CLOSE OUT-DATA.
                   ENTER RETURN-LINE WRTFIN.
                   ENTER COBOL WRTFIN.
               OPNCDS.
                   OPEN INPUT INCARDS.
                   ENTER FIXTBL, WITH INCARDS.
                   OPEN OUTPUT OUT-DATA.
                   ENTER FIXTBL, WITH OUT-DATA.
                   READ INCARDS AT END GO TO EOF-DTA.
                   READ INCARDS AT END GO TO EOF-DTA.
                   MOVE IN-REC TO HOLD-REC.
                   MOVE HOLD-ID TO REC-ID-HLD.
                   MOVE IN-REC TO TBL-LINE.
                   WRITE OUT-REC.
                   CLOSE OUT-DATA.
                   ENTER RETURN-LINE OPNCDS.
                   ENTER COBOL OPNCDS.
               EOF-DTA.
                   MOVE 1 TO DOLLAR-SWT.



                   GO TO D029-OUT.
       COBOL COMPILATION COMPLETED          TIME-00:00
#END


#SPURT   R PROG20

 ERROR CC  LOC    FFJKB YYYYY MU ML CARD   LABEL     STATEMENT                                   SM 70050
                                                     EDEF*FIXTBL
       00 00000  61000 00000               FIXTBL    ENTRY
          00001  12710 00000     00                  ENT*B7*L($-1)
          00002  12617 00000                         ENT*B6*L(B7)
          00003  11000 00322               LP2       ENT*A*322
          00004  54026 00005                         RSE*SET*U(B6+5)
          00005  10000 77766                         ENT*Q*77766
          00006  44026 00005                         RPL*LP*U(B6+5)
          00007  10000 00014     00                  PUT*SR1*U(B6+11D)
          00010  14026 00013
          00011  10000 00032     00                  PUT*SR2*L(B6+11D)
          00012  14016 00013
          00013  61010 00000     00                  EXIT
          00014  61000 00000               SR1       ENTRY
          00015  27400 00124                         SUB*Q*124*QZERO
          00016  61000 00021     00                  JP*SR1A
          00017  11000 00000                         CL*A
          00020  61010 00014     00                  EXIT
          00021  10000 00003               SR1A      #MSG  BFH ERROR
          00022  12700 00024     00
          00023  61000 00027     00
          00024  03223 01405
          00025  05071 31505
          00026  12272 72427
          00027  65020 00136
          00030  12000 20406
          00031  61010 00014     00                  EXIT
          00032  61000 00000               SR2       ENTRY
          00033  11000 00000                         CL*A
          00034  61010 00032     00                  EXIT
#END
#LOAD NMY PROG,DCPASS1
#END
```

MFOR Y PROG

FORTRAN IV COMPILATION 70222 SJ

```
C     MAIN OVERHAUL CALCULATION PROGRAM
      COMMON FIIN,FSC,NCC,MDLCD,UNISSU,RULSCD,COGSYM,APPCD,MNTCD,B,
     1 APPAC,APPAQ,PRPL,QNTSYS,CDMNCD,FSCM,SRCCD,ISUMPP,QNTAP,NUMBER,
     2 RRR,MRF,PCAP,RPF,JMDLCD,JCD,INP,IOUT,IHOLD,IMDLTP,IFNSTP,IFINAL,
     4 JTAP2R,ITAP1R,ITAP2R,NHOLD,N1R,N2R,NFIN,NMDLS,LBLK,AAA,AAB,AAC,
     5 NFINS,NYYY,  INSUR,AAD,AAE,AAF,OHLS,RRRQNT,ARPF,AMRF,APCAP,IQT,
     6 WEROUT,ITMTCD,COST,PL,SYSALC,OALC,TPP,T,NFINAL,NJ2,JHOLD,LCRTCD
      INTEGER FIIN(2),FSC,NCC,MDLCD(2),UNISSU,RULSCD(3),COGSYM
      INTEGER APPCD(10,4,2),MNTCD(2)    ,APPAC(10,2),APPAQ(10,2),PRPL(10)
      INTEGER QNTSYS,CDMNCD,B(2, 5),FSCM(2),SRCCD,ISUMPP,QNTAP(10,4)
      REAL MRF,NUMBER
      INTEGER JMDLCD(2,1000),JCD(2,10),IQT(1000)
      DIMENSION NUMBER(1000),RRR(10,4),MRF(10,4),PCAP(10,4),RPF(10)
C     AE=1, AT=2, AR=3, AP=4
      CALL INIT
      WRITE(IOUT,1052)
C     THIS LOOP READ ALL FIINS AND MATCHES THE QUANTITY TO THE FIIN
      DO 195 IJKL=1,NFINS
      CALL GETFIN




C     IF THIS ITEM IS A CONSUMABLE, JUST WRITE IT IN THE  OUTPUT FILE
      IF(RULSCD(3)-NYYY) 62,62,75
   62 CONTINUE
      WRITE(ITAP1R) FIIN,ITMTCD,FSC,NCC,FSCM,MDLCD,UNISSU,RULSCD,COGSYM,
     1  COST,PL,WEROUT,APPCD,MRF,RRR,PCAP,MNTCD,QNTAP,RPF,
     2         QNTSYS,CDMNCD,AAA,AAB,AAC,AAD,AAE,AAF,B,SRCCD,ISUMPP,
     3  AMRF,APCAP,ARPF,JCD,INSUR,LCRTCD
      N1R=N1R+1
      GO TO 195
   75 CONTINUE
C     CALCULATE THE AT APPCODES AND OUTPUT THE AT ITEMS TO IHOLD
      CALL ATCALC
C     END THE AT LOOP
  195 CONTINUE
   55 CONTINUE
      IF(NHOLD) 616,616,615
  615 CONTINUE
C     REWIND AND SWITCH UNITS
      REWIND IHOLD
      REWIND ITAP2R
```

```
      REWIND JTAP2R
      IT=ITAP2R
      ITAP2R=JTAP2R
      JTAP2R=IT
C   THIS IS THE MAIN AR NESTING LOOP
      DO 58 I=1,NHOLD
      READ(IHOLD,1010) JMDLCD(1,I),JMDLCD(2,I),NUMBER(I),IQT(I)
   58 CONTINUE
      NMDLS=NHOLD
      REWIND IHOLD
      NHOLD=0
C   SORT THE PREVIOUSLY PROCESSED APPCODES
      CALL SORT(JMDLCD,NMDLS,NUMBER,IQT)
      N2R=NJ2
      NJ2=0
      DO 60 JKL=1,N2R
      CALL ARCALC
   60 CONTINUE
  616 CONTINUE
      IF(NJ2) 79,79,951

  951 IF(NHOLD) 76,76,55
C   SHOULDNT EVER GET HERE--MEANS CIRCULAR NESTING OR SOMETHING SIMILAR
   76 WRITE(IOUT,1316)
 1316 FORMAT(1H0,4HOOPS)
      CALL ARERОR
   79 CONTINUE
      REWIND ITAP1R
      REWIND JHOLD
C   NOW THE CONSUMABLES
      NMDLS=NFINAL
      DO 700 IKL=1,NFINAL
      READ(JHOLD,1010) JMDLCD(1,IKL),JMDLCD(2,IKL),NUMBER(IKL),IQT(IKL)
  700 CONTINUE
      CALL SORT(JMDLCD,NMDLS,NUMBER,IQT)
      DO 800 IKL=1,N1R
      CALL APCALC
  800 CONTINUE
 1010 FORMAT(A4,A3,F15.5,I10)
 1050 FORMAT(10X,A4,A3,F15.5)
 1051 FORMAT(/(20X,A4,A3,I10))

 1052 FORMAT(1H1)
      WRITE(IOUT,1313) NFINAL
 1313 FORMAT(1H1,6HNFINAL, I10)
      REWIND 4
      WRITE(4) NFINAL
      STOP
      END
WEND
```

FORTRAN IV COMPILATION 70222 SJ

```
      SUBROUTINE ARENOR
      COMMON FIIN,FSC,NCC,MDLCD,UNISSU,RULSCD,COGSYM,APPCD,MNTCD,B,
     1 APPAC,APPAQ,PRPL,QNTSYS,CDMNCD,FSCM,SRCCD,ISUMPP,QNTAP,NUMBER,
     2 RRR,MRF,PCAP,RPF,JMDLCD,JCD,INP,IOUT,IHOLD,IMDLTP,IFNSTP,IFINAL,
     4 JTAP2R,ITAP1R,ITAP2R,NHOLD,N1R,N2R,NFIN,NMDLS,LBLK,AAA,AAB,AAC,
     5 NFINS,NYYY,  INSUR,AAD,AAE,AAF,OHLS,RRRQNT,ARPF,AMRF,APCAP,IQT,
     6 WEROUT,ITMTCD,COST,PL,SYSALC,OALC,TPP,T,NFINAL,NJ2,JHOLD,LCRTCD
      INTEGER FIIN(2),FSC,NCC,MDLCD(2),UNISSU,RULSCD(3),COGSYM
      INTEGER APPCD(10,4,2),MNTCD(2)    ,APPAC(10,2),APPAQ(10,2),PRPL(10)
      INTEGER QNTSYS,CDMNCD,B(2, 5),FSCM(2),SRCCD,ISUMPP,QNTAP(10,4)
      REAL MRF,NUMBER
      DIMENSION NUMBER(1000),RRR(10,4),MRF(10,4),PCAP(10,4),RPF(10)
      INTEGER JMDLCD(2,1000),JCD(2,10),IQT(1000)
      REWIND JTAP2R
      DO 800 I  =1,NJ2
       READ(JTAP2R) FIIN,ITMTCD,FSC,NCC,FSCM,MDLCD,UNISSU,RULSCD,COGSYM,
     1  COST,PL,WEROUT,APPCD,MRF,RRR,PCAP,MNTCD,QNTAP,RPF,APPAC,APPAQ,
     2  OHLS,QNTSYS,CDMNCD,AAA,AAB,AAC,AAD,AAE,AAF,B,SRCCD,ISUMPP,
     *  AMRF,APCAP,ARPF,JCD,INSUR,LCRTCD




      JMDLCD(1,I)=MDLCD(1)
      JMDLCD(2,I)=MDLCD(2)
  800 CONTINUE
      REWIND JTAP2R
      CALL SORT(JMDLCD,NJ2  ,NUMBER,IQT)
      DO 500 JKL=1,NJ2
       READ(JTAP2R) FIIN,ITMTCD,FSC,NCC,FSCM,MDLCD,UNISSU,RULSCD,COGSYM,
     1  COST,PL,WEROUT,APPCD,MRF,RRR,PCAP,MNTCD,QNTAP,RPF,APPAC,APPAQ,
     2  OHLS,QNTSYS,CDMNCD,AAA,AAB,AAC,AAD,AAE,AAF,B,SRCCD,ISUMPP,
     3  AMRF,APCAP,ARPF,JCD,INSUR,LCRTCD
  405 DO 408 I=1,10
      IF(APPCD(I,3,1)-LBLK) 407,406,407
  406 IF(APPCD(I,3,2)-LBLK) 407,408,407
  407 CALL SEARCH(JMDLCD,NJ2  ,APPCD(I,3,1),APPCD(I,3,2),INDX,IERROR)
      IF(IERROR) 415,408,415
  415 CONTINUE
      WRITE(IOUT,1614) FIIN       ,APPCD(I,3,1),APPCD(I,3,2)
 1614 FORMAT(1H0,5HFIIN ,A3,A4,20H HAS AN APPCODE TO      , A4,A3,
     1  35H WHICH IS NOT IN THE DATA PACKAGE    )
  408 CONTINUE
```

```
500 CONTINUE
    STOP 9999
    END
#END
```

```
#FOR Y PROG11
          F O R T R A N   I V   C O M P I L A T I O N                    70222 SJ
      SUBROUTINE INIT
      COMMON FIIN,FSC,NCC,MDLCD,UNISSU,RULSCD,COGSYM,APPCD,MNTCD,B,
     1 APPAC,APPAQ,PRPL,QNTSYS,COMNCD,FSCM,SRCCD,ISUMPP,QNTAP,NUMBER,
     2 RRR,MRF,PCAP,RPF,JMDLCD,JCD,INP,IOUT,IHOLD,IMDLTP,IFNSTP,IFINAL,
     4 JTAP2R,ITAP1R,ITAP2R,NHOLD,N1R,N2R,NFIN,NMDLS,LBLK,AAA,AAB,AAC,
     5 NFINS,NYYY,  INSUR,AAD,AAE,AAF,OHLS,RRRQNT,ARPF,AMRF,APCAP,IQT,
     6 WEROUT,ITMTCD,COST,PL,SYSALC,OALC,TPP,T,NFINAL,NJ2,JHOLD,LCRTCD
      INTEGER FIIN(2),FSC,NCC,MDLCD(2),UNISSU,RULSCD(3),COGSYM
      INTEGER APPCD(10,4,2),MNTCD(2)    ,APPAC(10,2),APPAQ(10,2),PRPL(10)
      INTEGER QNTSYS,COMNCD,B(2, 5),FSCM(2),SRCCD,ISUMPP,QNTAP(10,4)
      REAL MRF,NUMBER
      INTEGER JMDLCD(2,1000),JCD(2,10),IQT(1000)
      DIMENSION NUMBER(1000),RRR(10,4),MRF(10,4),PCAP(10,4),RPF(10)
C   IOUT= PRINTER
C   INP= CARD READER
C   IMDLTP = OUTPUT TAPE OF QTYSYS
C   IFNSTP= OUTPUT TAPE OF PASS 1
C   ITAP1R  = SCRATCH DATA SET FOR CONSUMABLE FIINS
C   ITAP2R = SCRATCH DATA SET FOR FIINS WITH AR APP CODES
```

```
C   IHOLD = SCRATCH DATA SET FOR SAVING MODEL CODE AND NO. OF OHLS
C   IFINAL = THE FINAL OUTPUT TAPE
      INTEGER BLANKS,NOYES
      DATA NOYES/4HNYYY/,BLANKS/4H    /
      NYYY=NOYES
      LBLK=BLANKS
      INP=1
      IOUT=2
      INUM=4
      IHOLD=9
      IFNSTP=5
      IFINAL=6
      JTAP2R=10
      JHOLD=11
      ITAP1R=7
      ITAP2R=8
      NFINAL=0
      NHOLD=0
      N1R=0
      N2R=0




      NJ2=0
C  THESE ARE AIRCRAFT PLANNING DATA


C  OALC IS THE NUMBER OF OVERHAULS PER AIRCRAFT


      READ(INP,1002) TPP,OALC,SYSALC
 1002 FORMAT(3F10.3)
C    NFINS IS OUTPUT BY THE MAIN FIIN PROGRAM
      REWIND JHOLD
      READ(INUM,1301) NFINS
 1301 FORMAT(2I10)
      RETURN
      END
#ENU
```

NFOR Y PROG12

F O R T R A N   I V   C O M P I L A T I O N   70222 SJ

```
      SUBROUTINE GETFIN
      COMMON FIIN,FSC,NCC,MDLCD,UNISSU,RULSCD,COGSYM,APPCD,MNTCD,B,
     1 APPAC,APPAQ,PRPL,QNTSYS,CDMNCD,FSCM,SRCCD,ISUMPP,QNTAP,NUMBER,
     2 RRR,MRF,PCAP,RPF,JMDLCD,JCD,INP,IOUT,IHOLD,IMDLTP,IFNSTP,IFINAL,
     4 JTAP2R,ITAP1R,ITAP2R,NHOLD,N1R,N2R,NFIN,NMDLS,LBLK,AAA,AAB,AAC,
     5 NFINS,NYYY,  INSUR,AAD,AAE,AAF,OHLS,RRRQNT,ARPF,AMRF,APCAP,IQT,
     6 WEROUT,ITMTCD,COST,PL,SYSALC,OALC,TPP,T,NFINAL,NJ2,JHOLD,LCRTCD
      INTEGER FIIN(2),FSC,NCC,MDLCD(2),UNISSU,RULSCD(3),COGSYM
      INTEGER APPCD(10,4,2),MNTCD(2)   ,APPAC(10,2),APPAQ(10,2),PRPL(10)
      INTEGER QNTSYS,CDMNCD,B(2, 5),FSCM(2),SRCCD,ISUMPP,QNTAP(10,4)
      REAL MRF,NUMBER
      DIMENSION NUMBER(1000),RRR(10,4),MRF(10,4),PCAP(10,4),RPF(10)
      INTEGER JMDLCD(2,1000),JCD(2,10),IQT(1000)
      INTEGER ONE,TWO,SIX,SEVEN,A,B
      DATA BCONST/1HB/
      DATA ONE/1H1/,TWO/1H2/,SIX/1H6/,SEVEN/1H7/,A/1HA/
       READ(IFNSTP) FIIN,ITMTCD,FSC,NCC,FSCM,MDLCD,UNISSU,RULSCD,COGSYM,
     1  COST,PL,WEROUT,APPCD,MRF,RRR,PCAP,MNTCD,QNTAP,RPF,APPAC,APPAQ,
     -       CDMNCD,B,SRCCD,ISUMPP,INSUR,LCRTCD




C     JCD IS THE APPCODES SAVED FOR PRINTOUT IN THE NEXT PROGRAM
      DO159 I=1,10
      JCD(1,I)=LBLK
  159 JCD(2,I)=LBLK
      QNTSYS=0
      ARPF=0
      AMRF=0
      OHLS=0
      APCAP=0
      RRRQNT=0
C   FIND A PRINTOUT VALUE FOR RPF,MRF,AND PC/AP
      DO 261 I=1,10
      IF(RPF(I)) 261,261,262
  261 CONTINUE
      GO TO 264
  262 ARPF=RPF(I)
  264 DO 266 I=1,4
      DO 265 J=1,10
      IF(MRF(J,I)) 265,265,268
  265 CONTINUE
```

```
  266 CONTINUE
      GO TO 269
  268 AMRF=MRF(J,I)
  269 CONTINUE
      DO 272 I=1,4
      DO 271 J=1,10
      IF(PCAP(J,I)) 271,271,273
  271 CONTINUE
  272 CONTINUE
      GO TO 275
  273 APCAP=PCAP(J,I)
  275 CONTINUE
C     FIRST TWO APPCODES ARE THE ALLOWANCE LIST ITEMS
      DO 207 I=1,10
      IF(APPAQ(I,2)-ONE) 204,208,204
  204 IF(APPAQ(I,2)-SIX) 205,208,205
  205 IF(APPAQ(I,2)-A) 207,208,207
  207 CONTINUE
      GO TO 211
  208 CONTINUE
      JCD(1,1)=APPAQ(I,1)
      JCD(2,1)=APPAQ(I,2)
  211 CONTINUE
      DO 220 I=1,10
      IF(APPAQ(I,2)-TWO) 212,219,212
  212 IF(APPAQ(I,2)-SEVEN) 213,219,213
  213 IF(APPAQ(I,2)-BCONST)220,219,220
  220 CONTINUE
      GO TO 225
  219 CONTINUE
      JCD(1,2)=APPAQ(I,1)
      JCD(2,2)=APPAQ(I,2)
  225 CONTINUE
C   GET ALL AVAILABLE (UP TO 10) APPCODES FOR PRINTOUT
      IX=2
      DO 67 J=1,4
      DO 66 I=1,10
      IF(APPCD(I,J,1)-LBLK) 65,74,65
   74 IF(APPCD(I,J,2)-LBLK) 65,66,65
   65 IX=IX+1
```

```
      JCD(2,IX)=APPCD(I,J,2)
      JCD(1,IX)=APPCD(I,J,1)
      IF(IX-10) 66,68,68
   66 CONTINUE
   67 CONTINUE
   68 CONTINUE
      DO 177 I=1,10
  173 IF(APPAC(I,1)-LBLK) 175,174,175
  174 IF(APPAC(I,2)-LBLK) 175,176,175
  175 IX=IX+1
      IF(IX-10) 176,177,177
  176 CONTINUE
      JCD(1,IX)=APPAC(I,1)
      JCD(2,IX)=APPAC(I,2)
  177 CONTINUE
      AAA=0
      AAB=0
      AAC=0
      AAD=0
      AAE=0
      AAF=0
      QNTSYS=0
   70 CONTINUE
      RETURN
      END
#END
```

F O R T R A N   I V   C O M P I L A T I O N   70222 SJ

```
      SUBROUTINE APCALC
      COMMON FIIN,FSC,NCC,MDLCD,UNISSU,RULSCD,COGSYM,APPCD,MNTCD,B,
     1 APPAC,APPAQ,PRPL,QNTSYS,CDMNCD,FSCM,SRCCD,ISUMPP,QNTAP,NUMBER,
     2 RRR,MRF,PCAP,RPF,JMDLCD,JCD,INP,IOUT,IHOLD,IMDLTP,IFNSTP,IFINAL,
     4 JTAP2R,ITAP1R,ITAP2R,NHOLD,N1R,N2R,NFIN,NMDLS,LBLK,AAA,AAB,AAC,
     5 NFINS,NYYY,  INSUR,AAD,AAE,AAF,OHLS,RRRQNT,ARPF,AMRF,APCAP,IQT,
     6 WEROUT,ITMTCD,COST,PL,SYSALC,OALC,TPP,T,NFINAL,NJ2,JHOLD,LCRTCD
      INTEGER FIIN(2),FSC,NCC,MDLCD(2),UNISSU,RULSCD(3),COGSYM
      INTEGER APPCD(10,4,2),MNTCD(2)   ,APPAC(10,2),APPAQ(10,2),PRPL(10)
      INTEGER QNTSYS,CDMNCD,B(2, 5),FSCM(2),SRCCD,ISUMPP,QNTAP(10,4)
      REAL MRF,NUMBER
      INTEGER JMDLCD(2,1000),JCD(2,10),IQT(1000)
      DIMENSION NUMBER(1000),RRR(10,4),MRF(10,4),PCAP(10,4),RPF(10)
      DATA D/1HD/
       READ(ITAP1R) FIIN,ITMTCD,FSC,NCC,FSCM,MDLCD,UNISSU,RULSCD,COGSYM,
     1  COST,PL,WEROUT,APPCD,MRF,RRR,PCAP,MNTCD,QNTAP,RPF,
     2        QNTSYS,CDMNCD,AAA,AAB,AAC,AAD,AAE,AAF,B,SRCCD,ISUMPP,
     3  AMRF,APCAP,ARPF,JCD,INSUR,LCRTCD
      OHLS=0
      RRRQNT=0
      AAE=0
      AAD=0
      AAC=0
      QNTSYS=0
      IF(ITMTCD-2) 710,710,600
  600 IF(CDMNCD-D) 610,620,610
  610 ITMTCD=2
      GO TO 710
  620 ITMTCD=1
  710 DO 720 I=1,10
      IF(APPCD(I,4,1)-LBLK) 712,711,712
  711 IF(APPCD(I,4,2)-LBLK) 712,720,712
  712 CONTINUE
      CALL SEARCH(JMDLCD,NMDLS ,APPCD(I,4,1),APPCD(I,4,2),INDX,IERROR)
      IF(IERROR) 719,713,719
  713 CONTINUE
      QNTSYS=QNTSYS+IQT(INDX)*QNTAP(I,4)
      AAE=AAE+NUMBER(INDX)*PCAP(I,4)*QNTAP(I,4)*RRR(I,4)
      AAD=AAD+IQT(INDX)*  QNTAP(I,4)*PCAP(I,4)*MRF(I,4)
```

```
      IF(ITMTCD-1) 718,718,714
C     BASE CONSUMABLE
  714 CONTINUE
  417 AAB=AAB+MRF(I,4)*        QNTAP(I,4)   *IQT(INDX)
 1010 FORMAT(1H0,3HAAB,F15.5,2X,3HAAD,F15.5,2X,3HAAE,F15.5,2X,3HQNT,I10)
      GO TO 718
  719 WRITE(IOUT,3716)APPCD(I,4,1),APPCD(I,4,2),FIIN(1),FIIN(2)
 3716 FORMAT(1H0,20HCOULDNT FIND APCODE ,A4,A3,10H FOR FIIN A3,A4)
  718  CONTINUE
  720 CONTINUE
      DO 807 I=1,10
      IF(APPCD(I,1,1)-LBLK) 800,801,800
  801 IF(APPCD(I,1,2)-LBLK) 800,807,800
  800 AAE=OALC*PCAP(I,1)*QNTAP(I,1)*RRR(I,1) + AAE
      QNTSYS=QNTSYS + QNTAP(I,1)
      AAD=AAD+ QNTAP(I,1)*PCAP(I,1)*MRF(I,1)
      IF(ITMTCD-1) 807,807,803
  803 AAB=AAB+ MRF(I,1)*QNTAP(I,1)
  807 CONTINUE
      WRITE(IFINAL) FIIN,ITMTCD,FSC,NCC,FSCM,MDLCD,UNISSU,RULSCD,COST,
     1 PL,WEROUT,QNTSYS,SRCCD,MNTCD,CDMNCD,ISUMPP,RRRQNT,B,AAA,AAB,AAD,
     2 AAE,AAF,AMRF,ARPF,APCAP,INSUR,JCD,LCRTCD,COGSYM
      NFINAL=NFINAL+1
      RETURN
      END
#END
```

```
      SUBROUTINE ARCALC
      COMMON FIIN,FSC,NCC,MDLCD,UNISSU,RULSCD,COGSYM,APPCD,MNTCD,B,
     1 APPAC,APPAQ,PRPL,QNTSYS,CDMNCD,FSCM,SRCCD,ISUMPP,QNTAP,NUMBER,
     2 RRR,MRF,PCAP,RPF,JMDLCD,JCD,INP,IOUT,IHOLD,IMDLTP,IFNSTP,IFINAL,
     4 JTAP2R,ITAP1R,ITAP2R,NHOLD,N1R,N2R,NFIN,NMDLS,LBLK,AAA,AAB,AAC,
     5 NFINS,NYYY,  INSUR,AAD,AAE,AAF,OHLS,RRRQNT,ARPF,AMRF,APCAP,IQT,
     6 WEROUT,ITMTCD,COST,PL,SYSALC,OALC,TPP,T,NFINAL,NJ2,JHOLD,LCRTCD
      INTEGER FIIN(2),FSC,NCC,MDLCD(2),UNISSU,RULSCD(3),COGSYM
      INTEGER APPCD(10,4,2),MNTCD(2)    ,APPAC(10,2),APPAQ(10,2),PRPL(10)
      INTEGER QNTSYS,CDMNCD,B(2, 5),FSCM(2),SRCCD,ISUMPP,QNTAP(10,4)
      REAL MRF,NUMBER
      INTEGER JMDLCD(2,1000),JCD(2,10),IQT(1000)
      DIMENSION NUMBER(1000),RRR(10,4),MRF(10,4),PCAP(10,4),RPF(10)
      READ(ITAP2R) FIIN,ITMTCD,FSC,NCC,FSCM,MDLCD,UNISSU,RULSCD,COGSYM,
     1 COST,PL,WEROUT,APPCD,MRF,RRR,PCAP,MNTCD,QNTAP,RPF,APPAC,APPAQ,
     2 OHLS,QNTSYS,CDMNCD,AAA,AAB,AAC,AAD,AAE,AAF,B,SRCCD,ISUMPP,
     3 AMRF,APCAP,ARPF,JCD,INSUR,LCRTCD
      IF(ITMTCD-4) 405,420,435
C     THIS IS A DEPOT REPAIRABLE
  405 DO 415 I=1,10
      IF(APPCD(I,3,1)-LBLK) 407,406,407
  406 IF(APPCD(I,3,2)-LBLK) 407,415,407
  407 CALL SEARCH(JMDLCD,NMDLS,APPCD(I,3,1),APPCD(I,3,2),INDX,IERROR)
      IF(IERROR) 415,408,415
  408 APPCD(I,3,1)=LBLK
      APPCD(I,3,2)=LBLK
      QNTSYS=QNTSYS+IQT(INDX)*QNTAP(I,3)
      OHLS=OHLS+NUMBER(INDX)*RRR(I,3)         *QNTAP(I,3)*PCAP(I,3)
      T=QNTAP(I,3)*MRF(I,3)*IQT(INDX)
      AAD=AAD+T*PCAP(I,3)
      AAB=AAB+T
  415 CONTINUE
      GO TO 450
C     THIS IS A BASE-DEPOT REPAIRABLE
  420 DO 428 I=1,10
      IF(APPCD(I,3,1)-LBLK) 422,421,422
  421 IF(APPCD(I,3,2)-LBLK)       422,428,422
  422 CALL SEARCH(JMDLCD,NMDLS,APPCD(I,3,1),APPCD(I,3,2),INDX,IERROR)
      IF(IERROR) 428,424,428
```

```
  424 T=QNTAP(I,3)*IQT(INDX)
      QNTSYS=QNTSYS+IQT(INDX)*QNTAP(I,3)
      AAB=AAB+T*MRF(I,3)
      AAA=AAA+T*ARPF
      AAD=AAD+T*PCAP(I,3)*MRF(I,3)
      AAF=AAF+T*ARPF*PCAP(I,3)
      OHLS=OHLS+NUMBER(INDX)*RRR(I,3)          *QNTAP(I,3)*PCAP(I,3)
      APPCD(I,3,1)=LBLK
      APPCD(I,3,2)=LBLK
  428 CONTINUE
      GO TO 450
C     THIS IS A BASE REPAIRABLE
  435 DO 440 I=1,10
      IF(APPCD(I,3,1)-LBLK) 438,437,438
  437 IF(APPCD(I,3,2)-LBLK) 438,440,438
  438 CALL SEARCH(JMDLCD,NMDLS,APPCD(I,3,1),APPCD(I,3,2),INDX,IERROR)
      IF(IERROR) 440,439,440
  439 APPCD(I,3,1)=LBLK
      APPCD(I,3,2)=LBLK
      T=IQT(INDX)*QNTAP(I,3)*ARPF
      QNTSYS=QNTSYS+IQT(INDX)*QNTAP(I,3)
      AAA=AAA+T
      AAF=AAF+T*PCAP(I,3)
      OHLS=OHLS+NUMBER(INDX)*RRR(I,3)          *QNTAP(I,3)*PCAP(I,3)
  440 CONTINUE
  450 CONTINUE
      DO 460 I=1,10
      IF(APPCD(I,3,1)-LBLK) 462,456,462
  456 IF(APPCD(I,3,2)-LBLK) 462,460,462
  460 CONTINUE
C     IF WE GET HERE, THERE ARE NO MORE AR APPCODES
      RRRQNT=OHLS*WEROUT
      WRITE(JHOLD,1010) MDLCD(1),MDLCD(2),OHLS,QNTSYS
      WRITE(IFINAL) FIIN,ITMTCD,FSC,NCC,FSCM,MDLCD,UNISSU,RULSCD,COST,
     1  PL,WEROUT,QNTSYS,SRCCD,MNTCD,CDMNCD,ISUMPP,RRRQNT,B,AAA,AAB,AAD,
     2  AAE,AAF,AMRF,ARPF,APCAP,INSUR,JCD,LCRTCD,COGSYM
      NFINAL=NFINAL+1
      WRITE(IHOLD,1010) MDLCD(1),MDLCD(2),OHLS,QNTSYS
 1010 FORMAT(A4,A3,F15.5,I10)
      NHOLD=NHOLD+1
```

```
      GO TO 500
  462 CONTINUE
      WRITE(JTAP2R) FIIN,ITMTCD,FSC,NCC,FSCM,MDLCD,UNISSU,RULSCD,COGSYM,
     1  COST,PL,WEROUT,APPCD,MRF,RRR,PCAP,MNTCD,QNTAP,RPF,APPAC,APPAQ,
     2  QHLS,QNTSYS,CDMNCD,AAA,AAB,AAC,AAD,AAE,AAF,B,SRCCD,ISUMPP,
     3  AMRF,APCAP,ARPF,JCD,INSUR,LCRTCD
      NJ2=NJ2+1
  500 CONTINUE
      RETURN
      END
#END
```

```
#FOR Y PRO15
      F O R T R A N   I V   C O M P I L A T I O N                      70222 SJ
       SUBROUTINE SEARCH(IARRAY,LX,IARG1,IARG2,NX,IERR )
C    THIS SUBROUTINE PICKS FROM A 2 DIMENSIONAL ARRAY IARRAY(2,LX),
C    AN INDEX NUMBER NX CORRESPONDING TO THE MATCH OF THE TWO ARGUMENTS
C    IARG1 AND IARG2.  AN ERROR CODE IS SET IF THERE WAS NO MATCH-IERR=1.
       DIMENSION IARRAY(2,100)
       IERR=0
        NX = LX/2 + MOD (LX,2)
       LSTLOW=0
       LSTHI=LX+1
     5 CONTINUE
       IF(LSTLOW-LSTHI) 41,30,30
    41 IF(IARRAY(1,NX)-IARG1) 1,6,3
     6 IF(IARRAY(2,NX)-IARG2) 1,2,3
     1 CONTINUE
       IF(NX-LSTLOW) 30,30,7
     7 CONTINUE
       LSTLOW=NX
       AL=LSTHI-LSTLOW
       NX=AL/2. + .6
```

```
      NX=NX+LSTLOW
      GO TO 5
    2 RETURN
    3 CONTINUE
      IF(NX-LSTHI) 8,30,30
    8 CONTINUE
      LSTHI=NX
      AL=LSTHI-LSTLOW
      NX=AL/2. + .6
      NX=NX+LSTLOW
      GO TO 5
   30 IERR=1
      RETURN
      END
#END
```

```
#FOR Y PRO156
        F O R T R A N   I V   C O M P I L A T I O N                          70222 SJ
      SUBROUTINE SORT(APPCD,LENGTH,FACTOR,IQTY)
C  THIS SUBROUTINE SORTS A 2 DIMENSIONAL INTEGER ARRAY CALLED APPCD FROM LOW
C  TO HIGH, THE LOWEST BEING FIRST, TREATING THE ARRAY AS DOUBLE
C  PRECISION, THE FIRST ELEMTNE BEING HIGHER. TWO ADDITIONAL ARRAYS,
C   ATTRIBUTES OF APPCD, ARE SWITCHED TO REFLECT CHANGES IN APPCD.
C  FACTOR IS FLOATING POINT, IQTY IS INTEGER.
      DIMENSION FACTOR(1)
      INTEGER APPCD(2,100),IQTY(1)
      IOUT=2
      IFLAG=0
      IFIRST=1
      ILAST=LENGTH-1
      IF(LENGTH-1) 550,550,501
  501 DO 510 I=IFIRST,ILAST
      J=I+1
  503 IF(APPCD(1,I)-APPCD(1,J)) 510,504,505
  504 IF(APPCD(2,I)-APPCD(2,J)) 510,510,505
  505 IT=APPCD(1,I)
      APPCD(1,I)=APPCD(1,J)
```

```
      APPCD(1,J)=IT
      IT=APPCD(2,I)
      APPCD(2,I)=APPCD(2,J)
      APPCD(2,J)=IT
      IT=IQTY(J)
      IQTY(J)=IQTY(I)
      IQTY(I)=IT
      T=FACTOR(J)
      FACTOR(J)=FACTOR(I)
      FACTOR(I)=T
      IFLAG=1
  510 CONTINUE
      M=ILAST-IFIRST
      IF(M) 550,550,512
  512 CONTINUE
      IFLAG=0
      DO 520 I=1,M
      J=ILAST-I
      K=J+1
      IF(APPCD(1,K)-APPCD(1,J)) 515,514,520
  514 IF(APPCD(2,K)-APPCD(2,J)) 515,520,520
  515 IT=APPCD(1,K)
      APPCD(1 ,K)=APPCD(1,J)
      APPCD(1,J)=IT
      IT=APPCD(2,J)
      APPCD(2,J)=APPCD(2,K)
      APPCD(2,K)=IT
      IT=IQTY(J)
      IQTY(J)=IQTY(K)
      IQTY(K)=IT
      T=FACTOR(K)
      FACTOR(K)=FACTOR(J)
      FACTOR(J)=T
      IFLAG=1
  520 CONTINUE
      ILAST=ILAST-1
      IFIRST=IFIRST+1
      IF(IFLAG) 550,550,530
  530 IFLAG=0
      GO TO 501
```

```
     550 RETURN
         END
#END

#FOR Y PRO117
         F O R T R A N   I V   C O M P I L A T I O N                                 70222 SJ
          SUBROUTINE ATCALC
          COMMON FIIN,FSC,NCC,MDLCD,UNISSU,RULSCD,COGSYM,APPCD,MNTCD,B,
         1 APPAC,APPAQ,PRPL,QNTSYS,CDMNCD,FSCM,SRCCD,ISUMPP,QNTAP,NUMBER,
         2 RRR,MRF,PCAP,RPF,JMDLCD,JCD,INP,IOUT,IHOLD,IMDLTP,IFNSTP,IFINAL,
         4 JTAP2R,ITAP1R,ITAP2R,NHOLD,N1R,N2R,NFIN,NMDLS,LBLK,AAA,AAB,AAC,
         5 NFINS,NYYY,  INSUR,AAD,AAE,AAF,OHLS,RRRQNT,ARPF,AMRF,APCAP,IQT,
         6 WEROUT,ITMTCD,COST,PL,SYSALC,OALC,TPP,T,NFINAL,NJ2,JHOLD,LCRTCD
          INTEGER FIIN(2),FSC,NCC,MDLCD(2),UNISSU,RULSCD(3),COGSYM
          INTEGER APPCD(10,4,2),MNTCD(2)   ,APPAC(10,2),APPAQ(10,2),PRPL(10)
          INTEGER QNTSYS,CDMNCD,B(2, 5),FSCM(2),SRCCD,ISUMPP,QNTAP(10,4)
          REAL MRF,NUMBER
          DIMENSION NUMBER(1000),RRR(10,4),MRF(10,4),PCAP(10,4),RPF(10)
          INTEGER JMDLCD(2,1000),JCD(2,10),IQT(1000)
          IFLAG=0
          IF(ITMTCD-4) 205,220,230
     C  DEPOT REPAIRABLE
      205 DO 210 I=1,10
          IF(APPCD(I,2,1)-LBLK) 207,206,207
      206 IF(APPCD(I,2,2)-LBLK) 207,210,207
```

```
  207 IFLAG=1
  209 T=MRF(I,2)*QNTAP(I,2)
      AAD=AAD+T*PCAP(I,2)
      AAB=AAB+T
      AAC=AAC+QNTAP(I,2)*RRR(I,2)*PCAP(I,2)
      QNTSYS=QNTSYS+QNTAP(I,2)
  210 CONTINUE
      OHLS=OHLS+ AAD*SYSALC*TPP*(1.-WEROUT)*PL*3./100.  +SYSALC*OALC*AAC
      GO TO 240
C   THIS ITEM IS A BASE-DEPOT REPAIRABLE
  220 DO 225 I=1,10
      IF(APPCD(I,2,1)-LBLK) 222,221,222
  221 IF(APPCD(I,2,2)-LBLK) 222,225,222
  222 IFLAG=1
      QNTSYS=QNTSYS+QNTAP(I,2)
      T=QNTAP(I,2)
      AAC=AAC+QNTAP(I,2)*RRR(I,2)*PCAP(I,2)
      AAD=AAD+T*MRF(I,2)*PCAP(I,2)
      AAA=AAA+T*ARPF
      AAB=AAB+T*MRF(I,2)




      AAF=AAF+T*ARPF*PCAP(I,2)
  225 CONTINUE
      OHLS=OHLS+ AAD*SYSALC*TPP*(1.-WEROUT)*PL*3./100.  +SYSALC*OALC*AAC
      GO TO  240
C   THIS ITEM IS A BASE REPAIRABLE
  230 DO 237 I=1,10
      IF(APPCD(I,2,1)-LBLK) 232,231,232
  231 IF(APPCD(I,2,2)-LBLK) 232,237,232
  232 IFLAG=1
      T=ARPF*QNTAP(I,2)
      AAA=AAA+T
      QNTSYS=QNTSYS+QNTAP(I,2)
      AAF=AAF+T*PCAP(I,2)
      AAC=AAC+QNTAP(I,2)*RRR(I,2)*PCAP(I,2)
  237 CONTINUE
      OHLS=OHLS+ AAD*SYSALC*TPP*(1.-WEROUT)*PL*3./100.  +SYSALC*OALC*AAC
C   ALL THREE REPAIRABLES COME HERE
C   IF THE KFLAG IS NOT SET POSITIVE, THIS IS NOT AN AT ITEM
  240 IF(IFLAG) 290,290,250
  250 CONTINUE
```

```
      RRRQNT=SYSALC*OALC*AAC*WEROUT
      WRITE(JHOLD,1010) MDLCD(1),MDLCD(2),OHLS,QNTSYS
      WRITE(IFINAL) FIIN,ITMTCD,FSC,NCC,FSCM,MDLCD,UNISSU,RULSCD,COST,
     1 PL,WEROUT,QNTSYS,SRCCD,MNTCD,CDMNCD,ISUMPP,RRRQNT,B,AAA,AAB,AAD,
     2 AAE,AAF,AMRF,ARPF,APCAP,INSUR,JCD,LCRTCD,COGSYM
      WRITE(IHOLD,1010) MDLCD(1),MDLCD(2),OHLS,QNTSYS
 1010 FORMAT(A4,A3,F15.5,I10)
      NHOLD=NHOLD+1
      NFINAL=NFINAL+1
      GO TO 300
  290 CONTINUE
      NJ2=NJ2+1
   59 CALL AECALC
      WRITE(JTAP2R) FIIN,ITMTCD,FSC,NCC,FSCM,MDLCD,UNISSU,RULSCD,COGSYM,
     1 COST,PL,WEROUT,APPCD,MRF,RRR,PCAP,MNTCD,QNTAP,RPF,APPAC,APPAQ,
     2 OHLS,QNTSYS,CDMNCD,AAA,AAB,AAC,AAD,AAE,AAF,B,SRCCD,ISUMPP,
     3 AMRF,APCAP,ARPF,JCD,INSUR,LCRTCD
  300 CONTINUE
      RETURN
      END
```

WFOR Y PRO118

FORTRAN IV COMPILATION 70222 SJ

```
      SUBROUTINE AECALC
      COMMON FIIN,FSC,NCC,MDLCD,UNISSU,RULSCD,COGSYM,APPCD,MNTCD,B,
     1 APPAC,APPAQ,PRPL,QNTSYS,CDMNCD,FSCM,SRCCD,ISUMPP,QNTAP,NUMBER,
     2 RRR,MRF,PCAP,RPF,JMDLCD,JCD,INP,IOUT,IHOLD,IMDLTP,IFNSTP,IFINAL,
     4 JTAP2R,ITAP1R,ITAP2R,NHOLD,N1R,N2R,NFIN,NMDLS,LBLK,AAA,AAB,AAC,
     5 NFINS,NYYY,  INSUR,AAD,AAE,AAF,OHLS,RRRQNT,ARPF,AMRF,APCAP,IQT,
     6 WEROUT,ITMTCD,COST,PL,SYSALC,OALC,TPP,T,NFINAL,NJ2,JHOLD,LCRTCD
      INTEGER FIIN(2),FSC,NCC,MDLCD(2),UNISSU,RULSCD(3),COGSYM
      INTEGER APPCD(10,4,2),MNTCD(2)     ,APPAC(10,2),APPAQ(10,2),PRPL(10)
      INTEGER QNTSYS,CDMNCD,B(2, 5),FSCM(2),SRCCD,ISUMPP,QNTAP(10,4)
      REAL MRF,NUMBER
      INTEGER JMDLCD(2,1000),JCD(2,10),IQT(1000)
      DIMENSION NUMBER(1000),RRR(10,4),MRF(10,4),PCAP(10,4),RPF(10)
C     THIS IS A REPAIRABLE-- DO THE  AE APPLICATIONS
      IF(ITMTCD-4) 79,86,95
C     MAKE SURE THIS IS A 5, NOT AN :UNKNOWN:
   95 IF(ITMTCD-5) 96,96,110
C    THIS IS A BASE  REPAIRABLE
   96 DO 98 I=1,10
      IF(APPCD(I,1,1)-LBLK) 116,97,116
   97 IF(APPCD(I,1,2)-LBLK) 116,98,116
  116 CONTINUE
      AAC=AAC+QNTAP(I,1)*PCAP(I,1)*RRR(I,1)
      QNTSYS=QNTSYS+QNTAP(I,1)
      AAA=AAA+ARPF*QNTAP(I,1)
      AAF=AAF+  ARPF*PCAP(I,1)*QNTAP(I,1)
   98 CONTINUE
      AAD=0
      GO TO 100
   79 CONTINUE
C     DEPOT REPAIRABLE
      DO 82 I=1,10
      IF(APPCD(I,1,1) -LBLK) 81,80,81
   80 IF(APPCD(I,1,2)-LBLK) 81,82,81
   81 T=       QNTAP(I,1)*MRF(I,1)
      QNTSYS=QNTSYS+QNTAP(I,1)
      AAB=AAB+T
      AAD=AAD+T*PCAP(I,1)
      AAC=AAC+      PCAP(I,1)*RRR(I,1)*QNTAP(I,1)
```

```
   82 CONTINUE
      GO TO 100
C     THIS IS A BASE-DEPOT REPAIRABLE
   86 DO 92 I=1,10
      IF(APPCD(I,1,1)-LBLK) 88,87,88
   87 IF(APPCD(I,1,2)-LBLK) 88,92,88
   88 CONTINUE
      T=QNTAP(I,1)
      QNTSYS=QNTSYS+QNTAP(I,1)
      AAA=AAA+T*ARPF
      AAB=AAB+T*MRF(I,1)
      AAF=AAF+T*PCAP(I,1)*ARPF
      AAD=AAD+T*PCAP(I,1)*MRF(I,1)
      AAC=AAC+T*PCAP(I,1)*RRR(I,1)
   92 CONTINUE
C     ALL THREE REPAIRABLES COME HERE TO CALCULATE OHLS AND RRRQNTITY
  100 CONTINUE
  110 CONTINUE
      OHLS=SYSALC*OALC*AAC +OHLS
      RRRQNT=WEROUT*OHLS



      RETURN
      END
#END
#LOAD NMY PROG.DCPASS2
#END
#OUT  D.JOB.DCPASS2
```

Preceding Page Blank - NOT FILMED

APPENDIX C

PROBABILITY-CONSTRAINT PROGRAM

The probability program consists of the job steps PROB1 and PROB2.

1. JOB STEP PROB1

The data inputs to PROB1 are FIIN, identification code, FSC, NCC, FSCM, model code, unit of issue, rules code, unit price, production lead time (PL), wearout rate (Z), applications, quantity per provisioning, source code, maintenance code, condemnation code, par pool quantity, RRR Quantity (RRRQ), nomenclature, part number, AAA, AAB, AAD, AAE, AAF, maintenance replacement factor, rotable pool factor, insurance quantity, local routing code, and cognizance symbol. These data, along with certain planning data, are used in computing average demands for spares. These average demands are used in job step PROB2 to compute optimal quantities of spares.

The planning data, which are read from cards, consist of the following:

- T1, the flying hours per month for consumable and wearout items
- T2, the flying hours per month for repairables
- T3, IMA TAT
- T4, the resupply time
- T5, the time to be protected
- T6, restockage time
- NBASE, the number of IOL columns desired
- ANAC, the number of flying hours that corresponds to each IOL column
- ANORS, the desired probability constraint to which the program will be run

Operating with the demand-floor option requires that for each type of item, a minimum expected demand and a period of time in months be read from a card. For any item, if the average demand for spares to be placed at an operating base over this period for a particular column is less than the minimum, the item is assigned 0 spares. An asterisk will be printed next to the zero to indicate that demand-floor action has taken place.

Average demands for operating base and backup spares, and average demands for a cutoff-criterion (demand floor) comparison are computed for every item except depot consumables. Depot consumables receive only backup spares. The average demand per month for operating-base spares and the average demand per month for the cutoff criterion are directly proportional to the flying-hour programs. Since there are several different flying-hour programs (one for each IOL column), a value proportional to average demand

demand (not including the flying-hour program for each column) is computed for each item; these values are then multiplied by each of the numbers of flying hours per month in turn to obtain average demands, resulting in a more efficient program operation. The average demands depend on the type of item and the repair location. The equations used in computing the proportionality constants for the six different item types are shown in Table C—1. For a description of the six different item types, see Subsection 2.2.3 of the text.

Table C—1. **PROPORTIONALITY CONSTANTS**

Item Type	Constant of Proportionality for Average Demand*	Constant of Proportionality for Cutoff-Criterion Comparison*	Average Demand for Backup Spares*
Base Depot Item	(AAA · T3 + AAB · T4)/3000	(AAA + AAB) · MO/100	[3AAF · Z · PL · T1 + AAD · (T2 · T6/30 + 3 · Z · PL · T1)]/ 100 + RRRQ
Base-Repairable Item	AAA · T3/3000	AAA · MO/100	3AAF · Z · PL · T1/ 100 + RRRQ
Depot-Repairable Item	AAB · T4/3000	AAB · MO/100	AAD · (T2 · T6/30 + 3Z · PL · T1)/100 + RRRQ
Base-Consumable Item	AAB · T5/3000	AAB · MO/100	AAD · PL · T1/33.3 + AAE · PL/6
Depot-Consumable Item			AAD · PL · T1/33.3 + AAE · PL/6

*See Appendix F for definitions.
MO is the number of months inputted to establish a demand-floor criterion for this type of item.

Depot-consumable items are considered only in computing backup spares. Therefore, no constants of proportionality for average demand or for the cutoff criterion are computed.

2. JOB STEP PROB2

PROB2 uses the average demands for spares computed in PROB1 to calculate optimal numbers of spares. The only card-input parameter is an integer that represents the number of items for which data can be kept in core simultaneously (a function of available memory). Since the program is intended to be capable of computing spares for any number of items, these data must be stored on a drum for every item; and only a limited number of items are treated at one time.

Optimal spares are computed for each column in turn. and then for the backup. The criterion for determining which item is to receive a spare is the value of the need cost factor,

$$VAL = \frac{1 - SUMPC}{COST}$$

where SUMPC is the probability of spares sufficiency for a particular item and COST is the unit price of that item. VAL is calculated for all items. The item for which VAL is largest receives a spare. The probability is computed by

$$SUMPC = \sum_{j=1}^{N} \frac{e^{AD} (AD)^j}{j!}$$

where N is the number of spares and AD is the average demand (for site spares, the product of the proportionality constant and the flying-hour program; and for backup, the average demand as computed in PROB1). The overall probability of sufficiency is the product of the probabilities for each item; and the calculations for each column and for the backup are complete when this quantity is at least as great as the desired probability-of-spares-sufficiency input to PROB1 (ANORS).

For the overall probability to be at least as great as ANORS, the probability of spares sufficiency for each item must be greater than or equal to ANORS. If 500 or fewer items are being treated, the probability of spares sufficiency (500 was determined as the most reasonable upper limit for the number of items that can be kept in memory core simultaneously) is insured by adding spares (increasing N in the equation above) to each item, in turn, until for that item SUMPC is greater than or equal to ANORS. To speed up this procedure for a large number of items it is desirable to add more spares immediately in this initial phase of the program. Therefore, if more than 500 items are being considered, a different method is used, as described below.

If M is the total number of items, then $(ANORS)^{1/M} \equiv P1$ is computed and one item is given enough spares so that its probability of sufficiency is at least as great as P1. For this item, with unit price = COST,

$$VAL_R = (1. - SUMPC)/COST$$

is computed, and every other item is given enough spares such that

$$(1. - SUMPC/COST) \leqslant VAL_R$$

After this phase of the program is completed, in which the items are treated individually, spares are added to the items, by using the iterative method mentioned above, until the overall probability of sufficiency is at least as great as ANORS. Since all the data necessary to compute SUMPC and VAL for every item for a large number of items cannot be kept in core simultaneously, these quantities are saved on drum, and only the data for the 500

largest VALs are kept in core. Spares are then added to the items, in turn, for which VAL is largest, with VAL being recomputed as a spare is added; and the overall probability of spares sufficiency is calculated. This process continues until the overall probability of sufficiency is at least ANORS or until every item has a VAL lower than the 501st item. If the overall probability of sufficiency is still less than ANORS, the items are reordered by VAL; and the data for the now-current 500 largest VALs are kept in core, and calculations proceed as before.

When enough spares have been added so that the overall probability of sufficiency is at least ANORS, for a particular column, the next column is treated; i.e., average demands are recomputed for a new flying-hour program, and optimum quantities of spares are calculated by the same method. When this process has been completed for every column, the same method is applied for backup spares.

After this process has been completed, the quantities and descriptive data, such as part number and nomenclature, are printed out by item. For base-depot items, spares are distributed between base and depot. In addition, a tape is created that includes the column spreads for repairables and base consumables. This tape is used in updating the Master Data File. Descriptive data are also printed for items whose source codes are P2. At the end of the printout a summary is included, indicating the overall probability of sufficiency per IOL column and for the backup column. Additionally, the cost of spares for each IOL column and for the backup is shown.

A program throughput is shown in Figure C—1. Program logic is shown in Figure C—2, which is followed by a complete program listing.

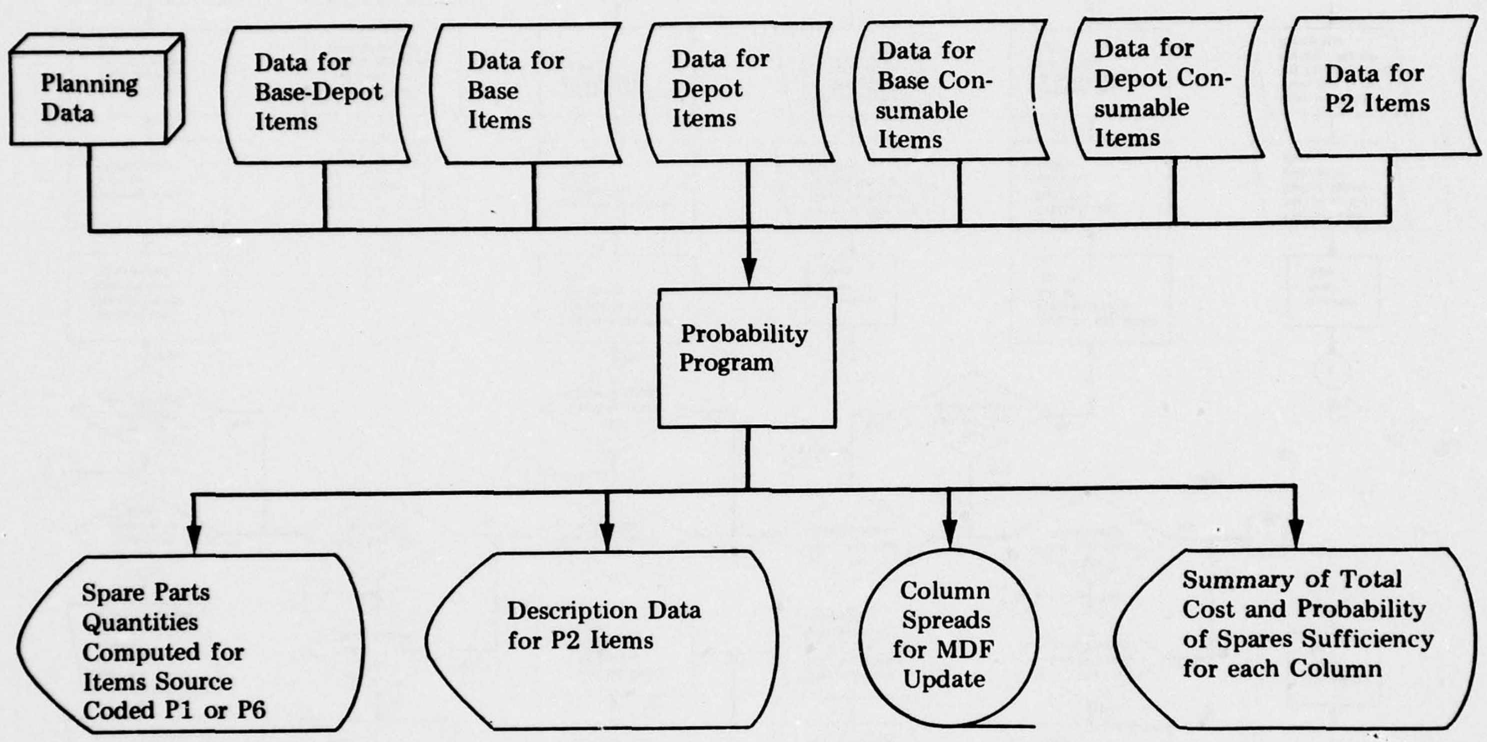


Figure C—1. **THROUGHPUT DESCRIPTION FOR PROBABILITY PROGRAM**

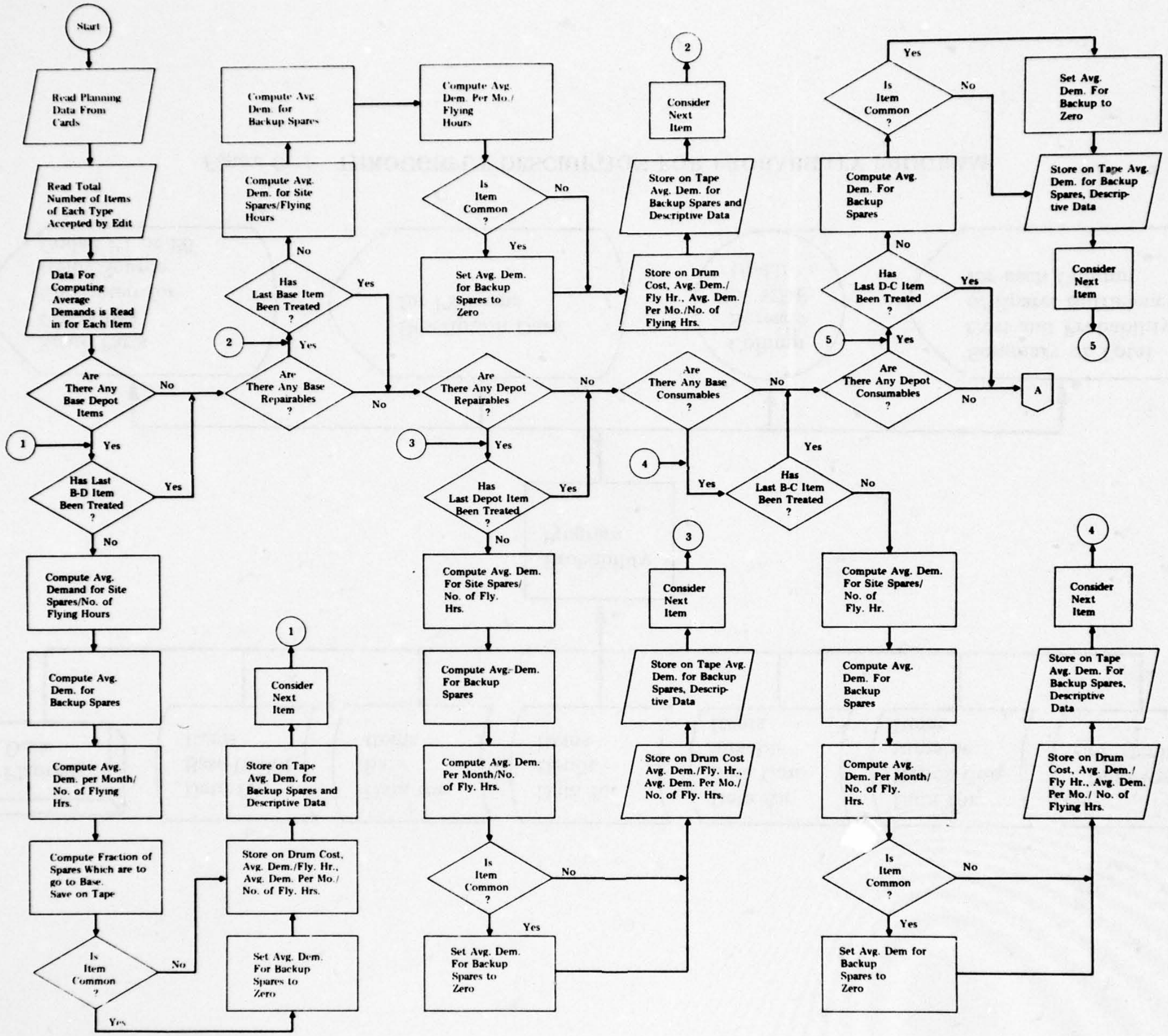


Figure C 2 FLOW CHART OF PROBABILITY-CONSTRAINT PROGRAM

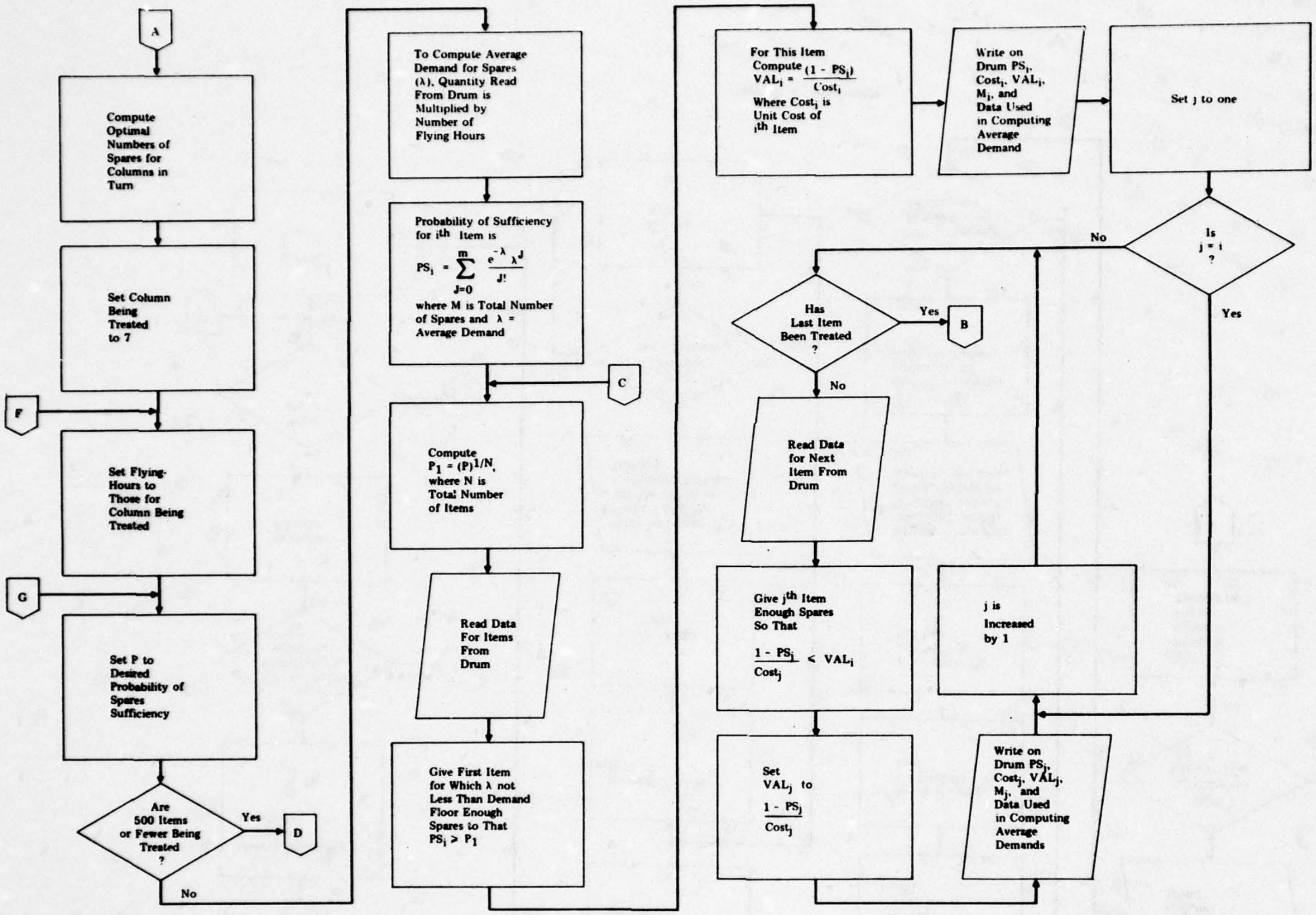


Figure C–2. (continued)

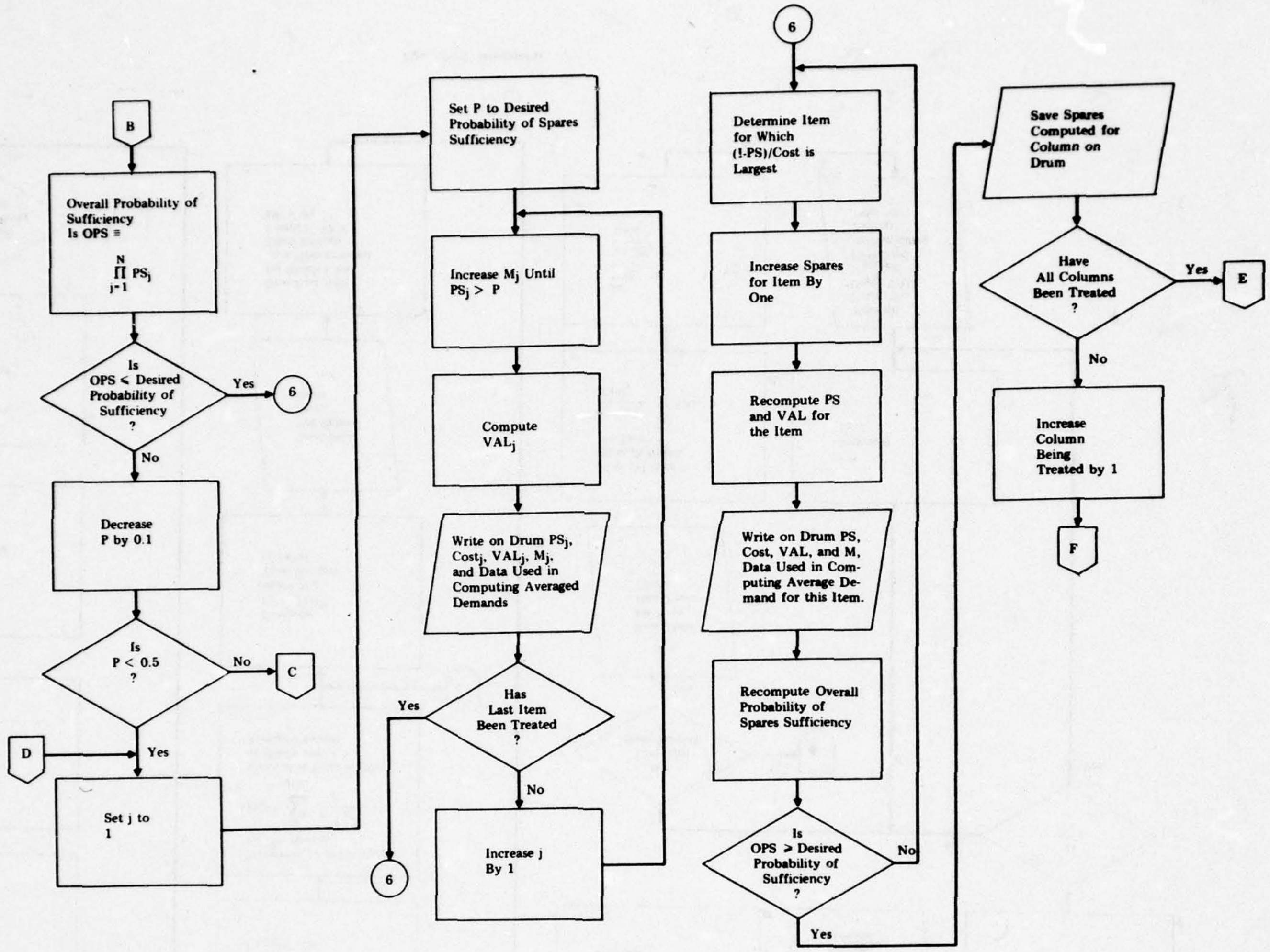


Figure C–2. **(continued)**

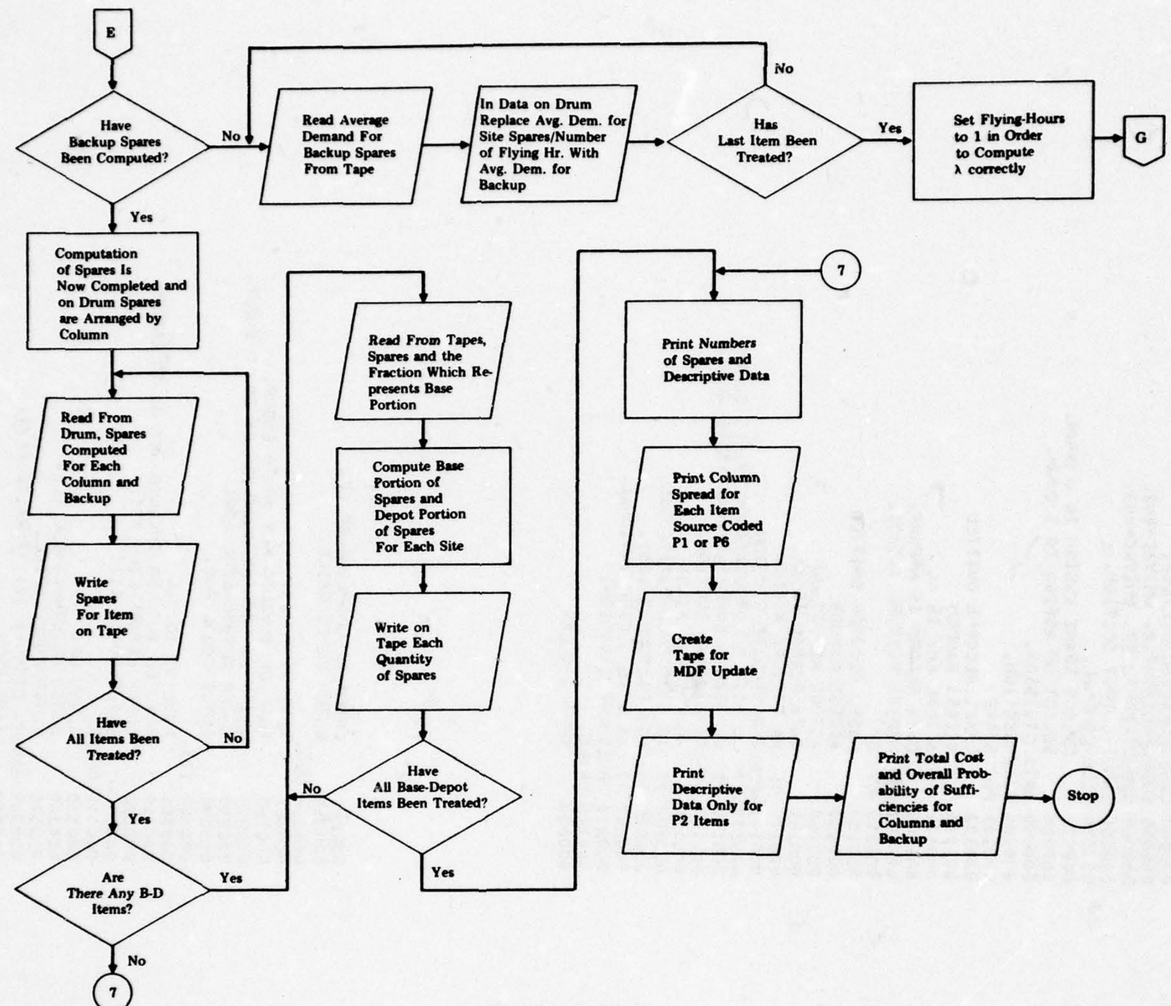


Figure C–2. (continued)

```
#COB BLVY OFILE1

     UNIVAC 490/491/492/494  COBOL COMPILATION   DATE-70050     TIME-00:00          VERGC

            000010 IDENTIFICATION DIVISION.
            000020 PROGRAM-ID. UNIVAC ASO.
            000030 ENVIRONMENT DIVISION.
            000040 CONFIGURATION SECTION.
            000050 SOURCE-COMPUTER. UNIVAC-492.
            000060 OBJECT-COMPUTER. UNIVAC-492.
            000061 INPUT-OUTPUT SECTION.
            000070 FILE-CONTROL.
            000075     SELECT SPARE ASSIGN TO J DRUM.
            000080     SELECT DF ASSIGN TO I DRUM.
            000090 DATA DIVISION.
            000100 FILE SECTION.
            000101 FD  SPARE
            000102     LABEL RECORDS OMITTED
            000103     ACCESS RANDOM
            000104     ACTUAL KEY IS LL
            000105     DATA RECORD IS NSPARE.
            000107 01  NSPARE PICTURE X(125).
            000110 FD  DF
            000130     LABEL RECORDS OMITTED
            000150     ACCESS RANDOM
            000160     ACTUAL KEY IS JJ
            000161     DATA RECORD IS AA.
            000170 01  AA PICTURE X(65).
            000171 WORKING-STORAGE SECTION.
            000172 77  JJ SIZE 5 COMPUTATIONAL VALUE ZERO.
            000175 77  LL SIZE 5 COMPUTATIONAL VALUE ZERO.
            000180 COMMON-STORAGE SECTION.
            000190 77  BB PICTURE X(65).
            000200 77  II SIZE 5 COMPUTATIONAL.
            000205 77  NS1 PICTURF X(125).
            000207 77  KK SIZE 5 COMPUTATIONAL.
            000210 PROCEDURE DIVISION.
            000220 OFILE.
            000230     OPEN I-O DF.

            000240     ENTER RETURN-LINE OFILE.
            000250     ENTER COBOL OFILE.
            000260 OOO.
            000261     MOVE II TO JJ.
            000265     READ DF INVALID KEY GO TO ENDUP.
            000270     WRITF AA FROM BB INVALID KEY GO TO ENDUP.
            000280     ENTER RETURN-LINE OOO.
            000290     ENTER COBOL OOO.
            000300 III.
            000301     MOVE II TO JJ.
            000310     READ DF INTO BB INVALID KEY GO TO ENDUP.
            000320     ENTER RETURN-LINE III.
            000330     ENTER COBOL III.
            000340 CFILE.
            000350     CLOSE DF.
            000360     ENTER RETURN-LINE CFILE.
            000370     ENTER COBOL CFILE.
            000380 ENDUP. DISPLAY II, 'INVALID KEY'.
            000390     STOP RUN.
            000400 OFILE1.
            000410     OPEN I-O SPARE.
            000420     ENTER RETURN-LINE OFILE1.
            000430     ENTER COBOL OFILE1.
            000440 ODRUM.
            000450     MOVE KK TO LL.
            000460     READ SPARE INVALID KEY GO TO ENDUP.
            000470     WRITF NSPAPE FROM NS1 INVALID KEY GO TO ENDUP.
            000480     ENTER RETURN-LINE ODRUM.
            000490     ENTER COBOL ODRUM.
            000500 IDRUM.
            000510     MOVE KK TO LL.
            000520     READ SPARE INTO NS1 INVALID KEY GO TO ENDUP.
            000522     ENTER RETURN-LINE IDRUM.
            000524     ENTER COBOL IDRUM.
            000530 CFILE1.
            000540     CLOSE SPARE.
            000550     ENTER RETURN-LINE CFILE1.
            000560     ENTER COBOL CFILE1.
      COBOL COMPILATION COMPLETED        TIME-00:00
#END
```

#FOR YX CALVAL

```
      F O R T R A N   I V   C O M P I L A T I O N                          70050 SJ
      SUBROUTINE CALVAL(ITOT,ANORS,ANAC,CTOFA,NTYPE,JVAR)
       COMMON PRECL,SUMPC,COST,Y,VAL,II,CT,I
      DIMENSION CTOFA(5),NTYPE(5)
      IO=2
      IF(ITOT-JVAR)650,650,10
 10   P1=ANORS
       K = 1
462    S = P1**(1./ITOT)
475    DO 400 I =K,ITOT
       CALL III
       XL = Y*ANAC
      EXPDEM=CT*ANAC
      DO 300 J=1,5
      IF(I-NTYPE(J))301,301,300
 300  CONTINUE
 301  CTOF=CTOFA(J)
      IF(EXPDEM-CTOF)400,400,401
401    IF (XL-30.) 405,400,400
400    CONTINUE



       WRITE(IO,302)
 302   FORMAT(29H ALL ITEMS BELOW DEMAND FLOOR)
       STOP
405     SUMPC = EXP(-XL)
        IREC = I
        PRECL = SUMPC
        II = 0
420     IF (SUMPC-S) 410,415,415
410     II = II+1
        PRECL = PRECL*XL/II
        IF (PRECL-1.E-70) 465,465,470
465       K = I + 1
        GO TO 475
470     SUMPC = SUMPC + PRECL
        GO TO 420
415     VAL = (1.-SUMPC)/COST
505     VALSAV = VAL
        CALL OOO
        PRODUC = SUMPC
        DO 425 I=1,ITOT
```

AD-A054 471 ARINC RESEARCH CORP ANNAPOLIS MD F/G 15/5
ADAPTATION OF A PROVISIONING MODEL FOR GENERAL-PURPOSE USE BY T--ETC(U)
MAY 70 F J JACOBY, D THOMPSON, B COHEN N00019-70-C-0027
UNCLASSIFIED 928-51-5-1055 NL

2 OF 2
AD
A054471

END
DATE
FILMED
6-78
DDC

```
      IF (I-IREC) 430,425,430
430   CALL III
      EXPDEM=CT*ANAC
      DO 350 J=1,5
      IF(1-NTYPE(J))351,351,350
350   CONTINUE
351   CTOF=CTOFA(J)
      IF(EXPDEM-CTOF)429,429,432
429   II = 0
      GO TO 427
432   VAL = VALSAV * COST
      XL = Y*ANAC
      IF(XL-200.)500,416,416
416   II=XL+1.645*SQRT(XL)+.838253
427   SUMPC = 1.
      PRECL = 1.
      VAL = 0.
      GO TO 490
500   SUMPC = EXP(-XL)
      PRECL = SUMPC



      II = 0
445     IF (1.-SUMPC-VAL) 435,435,440
440   II = II+ 1
      PRECL = PRECL*XL/II
      IF (PRECL-1.E-70) 480,480,485
480    VAL = 0.
      SUMPC = 1.
      II = II-1
      GO TO 490
485   SUMPC = SUMPC + PRECL
      GO TO 445
435   PRODUC = PRODUC*SUMPC
      VAL = (1.-SUMPC)/COST
490   CALL OOO
425   CONTINUE
600   IF(PRODUC)650,650,601
601   IF(PRODUC-ANORS) 450,455,455
455   P1 = P1-.1
      IF (P1-.05) 650,460,460
460   F2 = VALSAV
```

```
      S = P1**(1./ITOT)
      I = IREC
      CALL III
      XL = Y*ANAC
520   IF (SUMPC-S)515,519,510
510   IF (II) 519,519,516
516   SUMPC = SUMPC-PRECL
      PRECL = PRECL*II/XL
      II = II-1
      GO TO 520
515   II = II + 1
      PRECL = PRECL*XL/II
      SUMPC = SUMPC + PRECL
519   VAL = (1.-SUMPC)/COST
      IF (F2-VAL) 605,455,605
605   VALSAV = VAL
      CALL OOO
      PRODUC = SUMPC
      DO 525 I=1,ITOT
      IF (I-IREC) 530,525,530

530   CALL III
      IF (II) 524,524,526
526   VAL = VALSAV*COST
      XL = Y*ANAC
      EXL = EXP(-XL)
      IF (EXL-1.E-70) 525,525,541
541   IF (PRECL-1.E-70) 532,532,545
532   II = 0
      SUMPC = EXL
      PRECL = EXL
245   IF (1.-SUMPC-VAL) 536,536,240
240   II = II + 1
      PRECL = PRECL*XL/II
      IF (PRECL-1.E-70) 280,280,285
280   VAL = 0.
      SUMPC = 1.
      II = II- 1
      GO TO 590
285   SUMPC = SUMPC + PRECL
      GO TO 245
```

```
545     IF (1.-SUMPC-VAL) 540,536,535
540     IF (II) 536,536,546
546     SUMPC = SUMPC - PRECL
        PRECL = PRECL*II/XL
        II = II-1
        GO TO 545
535     II = II + 1
        PRECL = PRECL*XL/II
        SUMPC = SUMPC + PRECL
536     VAL = (1.-SUMPC)/COST
524     PRODUC = PRODUC*SUMPC
590     CALL 000
525     CONTINUE
        GO TO 600
650    ANORS1=ANORS
651    SUMLOG=0.
        DO 60I=1,ITOT
        CALL III
       DO 380 J=1,5
       IF(I-NTYPE(J))381,381,380

380    CONTINUE
381    CTOF=CTOFA(J)
       EXPDEM=CT*ANAC
       IF(EXPDEM-CTOF)6050,6050,382
6050   II=0
       SUMPC=1.
       VAL=0.
       GO TO 69
382     II = 0
        XL = ANAC*Y
       IF(XL-200.)700,70,70
70     II=XL+1.645*SQRT(XL)+.838253
        SUMPC = 1.
        PRECL = 1.
        VAL = 0.
        GO TO 69
700     SUMPC = EXP(-XL)
        PRECL = SUMPC
       VAL=(1.-SUMPC)/COST
        IF (SUMPC-ANORS1)61,69,69
```

```
61      II = II+1
        PRECL = PRECL*XL/II
        SUMPC = SUMPC + PRECL
        F2 = VAL
        VAL = (1.-SUMPC)/COST
        IF(F2-VAL)65,70,65
65      IF (SUMPC-ANORS1)61,69,69
69       CALL 000
60      SUMLOG=SUMLOG+ALOG10(SUMPC)
        SUMLOG=SUMLOG+70.
        IF(SUMLOG)451,451,450
451     ANORS1=ANORS1+.01
        IF(ANORS1-1.)651,652,652
652     WRITE(IO,653)
653     FORMAT($ UNDERFLOW WITH .99 PROB$)
        STOP
450      RETURN
         END
#END
```

```
#FOR YX PROBA
        F O R T R A N   I V   C O M P I L A T I O N                 70050 SJ
         COMMON PRECL,SUMPC,COST,Y,VAL,II,CT,I
          DIMENSION ANAC(10),  ISC(5),  RCDE(3),  ANOME(5),  PART(5),CTOF
        1    (5)
          EQUIVALENCE(ISC(1),IBD),(ISC(2),IB),(ISC(3),ID),
        1     (ISC(4),IC),(ISC(5),IDC)
          INTEGER CMNT(2)
           INTEGER FSC,UNIT,RCDE,CSRC,      CMACC,ANOME,PART,LRC,COGSYM
          INTEGER FIIN(2),FSCM(2),CODM(2),APPL(2,10)
          REAL MONTHS(5),MO
          DATA NZZZ/4HNZZZ/
          REWIND 10
          REWIND 11
          REWIND 12
          REWIND 13
          REWIND 14
          IO=2
C        READ LENGTHS OF EACH DATA AREA
          REWIND 9
          READ(9)ISC ,N6
```

```
      REWIND 9
      CALL OFILE
      REWIND 6
      REWIND 7
       READ(1,4) NBASE
       READ(1,5) T1,T2,T3,T4,T5,T6,ANORS
      READ(1,5)  (CTOF(I),MONTHS(I),I=1,5)
       READ(1,5) (ANAC(I),I=1,NBASE)
4     FORMAT(16I5)
 5    FORMAT(10F8.1  )
       IDS = 9
C     COUNT NUMBER OF ITEMS
       ITOT = IBD + IB + ID + IC
       JTOT = ITOT + IDC
       IF (JTOT) 450,451,450
450    I = 0
       WRITE(7) ISC,ANORS,CTOF,ANAC,ITOT,JTOT,NBASE  ,N6
       DO 8 J=1,5
       IDS = IDS + 1
       NUMREC = ISC(J)

       IF (NUMREC) 8,8,15
 15   MO=MONTHS(J)
       DO 7 K=1,NUMREC
       I = I + 1
       READ (IDS) FIIN,IDD,FSC,NCC,FSCM,CODM,UNIT,RCDE,COST,PL,Z,
     1    APPL,IQU,CSRC,CMNT,CMACC,IPPO,RRRQ,ANOME,PART,AAA,
     2    AAB,AAD,AAE,AAF,AMRF,RPF,    INSQTY,LRC,COGSYM
       CON1 = AAA * T3
       CON2 = AAB * T4
       CON3 = 3. * Z * PL * T1
C    IN THE CALCULATIONS WHICH FOLLOW Y IS A CONSTANT OF PROPORTIONALITY
C    FOR SITE SPARE EXPECTED DEMAND, CT IS A CONSTANT OF PROPORTIONALITY
C    FOR THE EXPECTED DEMAND CUTOFF CRITERION, ZZ IS THE EXPECTED DEMAND
C    FOR BACKUP SPARES
       GO TO (1001,1002,1003,1004,1005),J
C     BASE-DEPOT ITEM
 1001  Y=(CON1+CON2)  /3000.
       ZZ=(AAF*CON3+AAD*(T2*T6/30.+CON3))/100.+RRRQ
       CT=(AAA+AAB)*MO/100.
C    RAT IS PERCENTAGE OF SITE SPARES ASSIGNED TO DEPOT FOR BASE-DEPOT
```

```
C    ITEMS.  THE OTHER SPARES ARE THE BASE PORTION
          RAT = CON2/(CON1 + CON2)
          WRITE(7) FIIN,COST,ZZ,RAT,ANOME,FSC,APPL,FSCM,CODM,UNIT,CSRC,
      1     CMNT,CMACC,AMRF,RPF,IOU,    T3,RCDE,IPPQ,CT,NCC       ,PART
      2    ,INSQTY ,LRC,COGSYM
          GO TO 610
C    BASE ITEM
 1002  Y=CON1/3000.
          ZZ=AAF*CON3/100.+RRRQ
          CT=AAA*MO/100.
          GO TO 610
C     DEPOT ITEM
 1003  Y=CON2/3000.
          ZZ=AAD*(T2*T6/30.+CON3)/100.+RRRQ
          CT=AAB*MO/100.
          GO TO 610
C    CONSUMABLE ITEM
 1004  Y=AAB*T5/3000.
          CT=AAB*MO/100.
C    DEPOT CONSUMABLE
 1005  ZZ=AAD*PL*T1/33.333+AAE*PL/6.
610    IF(RCDE(1)-NZZZ)611,611,612
 611  ZZ=0.
 612   WRITE(6) FIIN,COST,ZZ,RAT,ANOME,FSC,APPL,FSCM,CODM,UNIT,CSRC,
      1     CMNT,CMACC,AMRF,RPF,IOU,    T3,RCDE,IPPQ,CT,NCC     ,PART,INSQTY
      2    ,LRC,COGSYM
          IF(1-ITOT)2,2,7
   2   CALL 000
7      CONTINUE
8       CONTINUE
          CALL CFILE
          REWIND 6
          REWIND 7
          REWIND 10
451     STOP
          END
*END
```

WFOR YX PROBB


```
      COMMON PRECL,SUMPC,COST,Y,VAL,II,CT,I,IIZ0(25),ISPARE
      DIMENSION IND(10)
       DIMENSION ANAC(10),  ISC(5),  RCDE(3),  ANOME(5),  PART(5),
     1   LLL(5),
     1   PBA(10),NSPARE(25,11),NS(11),MM(11),NUMBER(6),ZCOST(11)
      DIMENSION AVAL(500),ASUMPC(500),APRECL(500),ACOST(500),AY(500),
     1   ISP(500),IPOINT(500),ACT(500)
      DIMENSION NTYPE(5),CTOF(5),CSAVE(5),CTOFF(6)
       EQUIVALENCE(ISC(1),IBD)
      INTEGER CMNT(2) ,X,X1(2)
       INTEGER FSC,UNIT,RCDE,CSRC,     CMACC,ANOME,PART,LRC,COGSYM
      INTEGER FIIN(2),FSCM(2),CODM(2),APPL(2,10)
      INTEGER AC,A,C
      DATA IBLNK/1H /,IAST/1H*/,NZZZ/4HNZZZ/,A/1HA/,C/1HC/,IBLNKS/4H
     1/,X/4HXXXX/,X1(1)/1H1/,X1(2)/1H2/
      REWIND 5
      REWIND 6
      REWIND 7
      REWIND 8


       READ(7) ISC,ANORS,CTOF,ANAC,ITOT,JTOT,NBASE              ,N6
      DO 1500 J=1,NBASE
      PBA(J)=0.
 1500 ZCOST(J.)=0.
      READ(1,1492)JVARO
 1492  FORMAT(I5)
      IP=5
      IO=2
      ISPARE=0
      CALL OFILE
      CALL OFILE1
      NTYPE(1)=ISC(1)
      NTYPE(2)=ISC(2)+NTYPE(1)
      NTYPE(3)=ISC(3)+NTYPE(2)
      NTYPE(4)=ITOT
      NTYPE(5)=JTOT
       IF (JTOT) 450,451,450
 450   IF (ITOT) 580,580,449
 449   IZ3 = 0
 582   DO 10 J=1,NBASE
```

```
      ANACJ=ANAC(J)
      IF(JVARO-ITOT)913,913,590
590   JVAR=ITOT
      GO TO 915
913   JVAR=JVARO
915   CALL CALVAL(ITOT,ANORS,ANACJ,CTOF,NTYPE,JVAR)
      IOUT=0
      SS=1.
      DO 896 I=1,ITOT
      CALL III
896   SS=SS*SUMPC
      IF(SS-ANORS)914,12,12
914   DO 100 I=1,JVAR
      CALL III
      ACT(I)=CT
      AVAL(I)=VAL
      ASUMPC(I)=SUMPC
      APRECL(I)=PRECL
      AY(I)=Y
      ACOST(I)=COST



      ISP(I)=II
100   IPOINT(I)=I
      JVAR1=JVAR+1
      IF(JVAR1-ITOT)622,622,623
622   MIN=0
      DO 93 I=JVAR1,ITOT
      IF(MIN)494,493,494
493   AMIN=AVAL(1)
      J1=1
      DO 491 I1=2,JVAR
      IF(AVAL(I1)-AMIN)492,492,491
492   J1=I1
      AMIN=AVAL(I1)
491   CONTINUE
      MIN=1
494   CALL III
      IF(VAL-AMIN)93,93,94
94    AVAL(J1)=VAL
      ACT(J1)=CT
      ASUMPC(J1)=SUMPC
```

```
      APRECL(J1)=PRECL
      ACOST(J1)=COST
      AY(J1)=Y
      ISP(J1)=II
      IPOINT(J1)=I
      MIN=0
93    CONTINUE
      XMAX1=AVAL(1)
      JVAR1=1
      DO 96 I=2,JVAR
      IF(AVAL(I)-XMAX1)97,96,96
97    XMAX1=AVAL(I)
      JVAR1=I
96    CONTINUE
      GO TO 392
623   XMAX1=0.
      JVAR1=ITOT+1
392   IVAR2=JVAR1+1
      IVAR1=JVAR1-1
92    XMAX=XMAX1

      J1=0
      IF(IVAR1)191,193,191
191   DO 190 I=1,IVAR1
      IF(AVAL(I)-XMAX)190,192,192
192   XMAX=AVAL(I)
      J1=I
190   CONTINUE
193   IF(IVAR2-JVAR)196,196,197
196   DO 195 I=IVAR2,JVAR
      IF(AVAL(I)-XMAX)195,198,198
198   XMAX=AVAL(I)
      J1=I
195   CONTINUE
197   IF(J1)390,390,295
390   DO 391 K=1,JVAR
      I=IPOINT(K)
      CT=ACT(K)
      PRECL=APRECL(K)
      SUMPC=ASUMPC(K)
      COST=ACOST(K)
```

```
      VAL=AVAL(K)
      Y=AY(K)
      II=ISP(K)
391   CALL 000
      IF(IOUT)914,914,12
295   IF(APRECL(J1)-1.E-70)323,323,322
323   SUMPC1=ASUMPC(J1)
      ASUMPC(J1)=1.
      AVAL(J1)=0.
        GO TO 261
322   ISP(J1)=ISP(J1)+1
      APRECL(J1)=APRECL(J1)*AY(J1)*ANACJ  /ISP(J1)
      SUMPC1=ASUMPC(J1)
      ASUMPC(J1)=ASUMPC(J1)+APRECL(J1)
      AVAL(J1)=(1.-ASUMPC(J1))/ACOST(J1)
      IF(XMAX-AVAL(J1))510,510,261
510   ISP(J1)=ISP(J1)-1
      GO TO 323
261   SS=SS*ASUMPC(J1)/SUMPC1
      IF(SS-ANORS)92,912,912




912   IOUT=1
      GO TO 390
12    I=0
      IF(IZ3)400,400,401
400   INUM=4
      GO TO 403
401    INUM = 5
403   DO 250 KKK=1,INUM
      IIZ2=ISC(KKK)
      IF(IIZ2)250,250,200
200   IF(IIZ2-25)201,201,202
201   IIREC=IIZ2
      GO TO 203
202   IIREC=25
203   DO 210 I5=1,IIREC
      I=I+1
      CALL III
210   IIZ0(I5)=II
      ISPARE=ISPARE+1
      CALL ODRUM
```

```
      IIZ2=IIZ2-IIREC
      IF(IIZ2)250,250,200
 250  CONTINUE
      PBA(J)=SS
 10   CONTINUE
      IF(IZ3)580,580,581
 580  IZ3=1
      TEMP=PBA(1)
      ITEM1=ITOT
      ITOT=JTOT
      NTEMP=NBASE
      NBASE=1
       ASAVE = ANAC(1)
      ANAC(1)=1.
      DO 360 I=1,5
      CSAVE(I)=CTOF(I)
 360  CTOF(I)=-10.
      DO 910 I=1,ITOT
       READ(6) FIIN,COST,Y
 910  CALL OOO
```

```
      REWIND 6
      GO TO 582
 581  PBU=PBA(1)
      PBA(1)=TEMP
      ITOT=ITEM1
      NBASE=NTEMP
       ANAC(1) = ASAVE
      DO 365 I=1,5
 365  CTOF(I)=CSAVE(I)
      CALL CFILE
      NB1=NBASE+1
      IIITOT=0
      DO 770 I=1,5
 770  LLL(I)=0
       DO 775 I=1,5
      IF(ISC(I))775,775,776
 776  JJ=ISC(I)/25
      IF(JJ*25-ISC(I))714,739,739
 714  JJ=JJ+1
 739  IF(I-4)41,41,725
```

```
   41  IIITOT=IIITOT+JJ
  725  LLL(I)=JJ
  775  CONTINUE
       DO 717 J=1,11
  717  ZCOST(J)=0.
        DO 715 J=1,5
       IF(ISC(J))715,715,785
  785  K=0
       IF(J-1)733,734,733
  733  JJ=J-1
       DO 745 III=1,JJ
  745  K=K+LLL(III)
  734  IZ=LLL(J)
       IF(J-4)731,731,732
  731  DO 755 LL=1,IZ
       IFACT=0
       K=K+1
       DO 753  LL1=1,NB1
       ISPARE=K+IFACT*IIITOT
       IFACT=IFACT+1




       CALL IDRUM
       DO 753 IZ1=1,25
  753  NSPARE(IZ1,LL1)=IIZO(IZ1)
       IF(LL-IZ)757,756,757
  756  I25=ISC(J)-25*(LL-1)
       GO TO 797
  757  I25=25
  797  DO 755 IZ1=1,I25
  755  WRITE(8     )(NSPARE(IZ1,LL1),LL1=1,NB1)
       GO TO 715
  732  K1=NBASE*IIITOT
       DO 795 LL=1,IZ
       K=K+1
       ISPARE=K+K1
       CALL IDRUM
       IF(LL-IZ)781,782,782
  781  I25=25
       GO TO 791
  782  I25=ISC(J)-25*(LL-1)
  791  DO 795 I=1,I25
```

```
795  WRITE(8    ) IIZ0(I  )
715  CONTINUE
     CALL CFILE1
     REWIND8
     REWIND 10
     REWIND 11
     IF(IBD)950,950,951
951  DO 920 J=1,IBD
     READ(8     )(NS(I),I=1,NB1)
      READ(7) FIIN,COST,ZZ,RAT
     MM(NB1)=NS(NB1)
     DO 923 I=1,NBASE
     XII=NS(I)*RAT
     MM(I)=XII
     IF(XII-MM(I))922,923,922
922  MM(I)=MM(I)+1
923  NS(I)=NS(I)-MM(I)
     WRITE( 10  )(MM(I),I=1,NB1)
920  WRITE(11    )(NS(I),I=1,NB1)
     REWIND 7




      READ(7)
     REWIND 10
     REWIND 11
950  NUMBER(1)=ISC(1)
     NUMBER(2)=ISC(2)
     NUMBER(3)=ISC(1)
     NUMBER(4)=ISC(3)
     NUMBER(5)=ISC(4)
     NUMBER(6)=ISC(5)
     CTOFF(1)=CTOF(1)
     CTOFF(2)=CTOF(2)
     CTOFF(3)=CTOF(1)
     CTOFF(4)=CTOF(3)
     CTOFF(5)=CTOF(4)
     CTOFF(6)=CTOF(5)
     IF(NBASE-10)736,737,737
736  KPUNCH=0
     GO TO 738
737  KPUNCH=1
738   DO 720 I=1,6
```

```
724  JJ=NUMBER(I)
     GO TO (761,762,763,764,765,766),I
761  WRITE(IO,30)
     WRITE(IO,71)
71   FORMAT(1H ,18HROTABLE POOL ITEMS,//,1H ,25HBASE-DEPOT (BASE PORTIO
    1N),/)
1996 NT=6
     NT1=11
     J2=2
     GO TO 80
762  WRITE(IO,72)
72   FORMAT(1H0,15HBASE REPAIRABLE,/)
     J2=J2+1
     GO TO 800
763  WRITE(IO,30)
     WRITE(IO,773)
773  FORMAT(1H ,15HATTRITION ITEMS,//,1H ,26HBASE-DEPOT (DEPOT PORTION)
    1,/)
722  NT=7
     NT1=10
     J2=2
     GO TO 80
764  WRITE(IO,74)
74   FORMAT(1H0,16HDEPOT REPAIRABLE,/)
     J2=J2+1
     GO TO 800
765  WRITE(IO,75)
75   FORMAT(1H0,15HBASE CONSUMABLE,/)
     J2=J2+1
     GO TO 800
766  WRITE(IO,30)
     WRITE(IO,76)
76   FORMAT(1H ,19HSYSTEM STOCKS ITEMS,/,1H0,16HDEPOT CONSUMABLE,/)
     J2=2
800  NT=6
     NT1=8
80   DEM=CTOFF(I)
     IF (JJ) 720,720,7000
7000 DO 721 J=1,JJ
     J2 = J2 + 1
```

```
      READ(NT) FIIN,COST,ZZ,RAT,ANOME,FSC,APPL,FSCM,CODM,UNIT,CSRC,
     1   CMNT,CMACC,AMRF,RPF,IQU,    T3,RCDE,IPPQ,CT,NCC    ,PART,INSQTY
     2 , LRC,COGSYM
      IF(J2-12)729,729,726
726   J2=1
      WRITE(IO,30)
30    FORMAT(1H1,2X,4HFIIN,5X,12HNOMENCLATURE,15X,3HFSN,13X,15HREFERENCE
     1 NO.  ,6X,              36HFSCM  MODEL   UNIT SRCE MAINT   MACC,/,1
     1H ,82X,29HCODE   ISSU CDE  CODE    CODE,/,1H ,9X,56HMRF       RPF
     1  QUAN/  UNIT PRICE   LRC  TAT     RULES,7X,59HPAR    BU    7
     1 8    9   10   11   12   13   14   15   16,/,1H ,26X,4HPROV,22X,
     2  3HIMA,5X,4HCODE,8X,3HPLQ/15H   APPLICATIONS,85X,8H INS QTY//)
729   IF(I-5)150,150,151
150   READ(NT1    )(NS(K1),K1=1,NB1)
      DO 1150 K1=1,NBASE
      IF(CT*ANAC(K1)-DEM)1151,1151,1152
1151  IND(K1)=IAST
      GO TO 1150
1152  IND(K1)=IBLNK
1150  CONTINUE




      WRITE(IO,33) FIIN,ANOME,FSC,    FIIN,PART,     FSCM,CODM,UNIT,
     1   CSRC,CMNT,CMACC,AMRF,RPF,IQU,COST,LRC,T3,RCDE,IPPQ,
     2 NS(NB1),(NS(K1),IND(K1),K1=1,NBASE)
33    FORMAT(1H ,A3,A4,2X,5A4,5X,A4,1H-,A3,1H-,A4,4X,5A4,2X,A4,A1,
     1 1X,A4,A3,2X,A2,3X,A2,4X,2A1,7X,A1/
     2       4X,2(3X,F7.3),I8,F12.2,2X,A5,F6.0,2X
     3 A4,A2,A4   ,1X,I5,1X,   I6,10(I4,A1)    )
      WRITE(IO,34)APPL     ,INSQTY
      DO 110 K1=1,NBASE
110   ZCOST(K1)=ZCOST(K1)+COST*NS(K1)
373   IF(RCDE(1)-NZZZ)350,350,351
350   AC=C
      GO TO 1649
351   AC=A
1649  IF(I-3)1651,1650,1650
1650  K2=1
      GO TO 1652
1651  K2=2
1652  IF(APPL(1,K2)-IBLNKS)352,1653,352
1653  APPL(1,K2)=X
```

```
      APPL(2,K2)=X1(K2)
 352  IF(KPUNCH)374,374,376
 374  WRITE(IP,375)COGSYM
     1  ,NCC,FIIN,(APPL(K1,K2),K1=1,2),AC,(NS(K1),K1=1,NBASE)
 375  FORMAT( A2    ,A2,A3,A4,1X,5HAD009,2X,A4,A3,3X,4HD005,1X,A1,9I4)
      GO TO 153
 376  WRITE(IP,377) COGSYM, NCC, FIIN, (APPL(K1,K2),K1=1,2), AC,
     1  (NS(K1),K1=1,9), COGSYM,NCC,FIIN, (APPL(K1,K2),K1=1,2), NS(10)
 377  FORMAT( A2    ,A2,A3,A4,1X,5HAD009,2X,A4,A3,3X,4HD005,1X,A1,9I4/
     1   2A2,A3,A4,1X,5HAD009,2X,A4,A3,3X,4HC007,1X,I4)
      GO TO 153
 151  READ(8    )NS(NB1)
       WRITE(IO,33) FIIN,ANOME,FSC,     FIIN,PART,     FSCM,CODM,UNIT,
     1     CSRC,CMNT,CMACC,AMRF,RPF,IQU,COST,LRC,T3,RCDE,IPPQ,
     2  NS(NB1)
      WRITE(IO,34)APPL     ,INSQTY
 34   FORMAT(1X,10(A4,A3,2X),8X,I7/)
 153  ZCOST(11)=ZCOST(11)+COST*NS(NB1)
 721  CONTINUE
 720  CONTINUE


      IF(N6)1560,1560,1550
C    PRINT P2 ITEMS
 1550  REWIND 15
      WRITE(IO,30)
      WRITE(IO,79)
 79   FORMAT(9H P2 ITEMS/)
      J2=2
      DO 1551 K1=1,N6
      READ(15) FIIN,IDC,FSC,NCC,FSCM,CODM,UNIT,RCDE,COST,PL,Z,APPL,IQU,
     1    CSRC,CMNT,CMACC,IPPQ,RRR,ANOME,PART,AAA,AAA,AAA,AAA,AAA,AMRF,
     1    RPF ,INSQTY,LRC
      J2=J2+1
      IF(J2-12)1553,1553,1552
 1552  J2=1
      WRITE(IO,30)
 1553  WRITE(IO,33) FIIN,ANOME,FSC,     FIIN,PART,     FSCM,CODM,UNIT,
     1     CSRC,CMNT,CMACC,AMRF,RPF,IQU,COST,LRC,T3,RCDE,IPPQ
 1551  WRITE(IO,34)APPL,INSQTY
      REWIND 15
 1560  WRITE(IO,90)PBU,(PBA(J),J=1,NBASE)
```

```
90      FORMAT(///,1H ,11F11.3)
        WRITE(10,750)ZCOST(11),(ZCOST(J),J=1,NBASE)
750     FORMAT(1H0,11F11.0)
        REWIND 5
        REWIND 6
        REWIND 7
        REWIND 8
        REWIND 10
451     STOP
        END
#END
#LOAD NMY EDIT1,EDIT
#END
#LOAD NMY PROBA,PROB1
#END
#LOAD NMY PROBB,PROB2
#END
```

APPENDIX D

COST-CONSTRAINT PROGRAM

The cost-constraint program (see Figure D—1) consists of the job steps COST 1 and COST 2.

1. JOB STEP COST 1

The input to COST 1 consists of a one-letter parameter (C or R, which indicates whether the run is for consumables or for repairables), the total number of bases being considered, a column entry associated with each individual base (the IOL column assigned for spares determination for that base), the cost constraint, and a tolerance within which the best probability level relative to the number of dollars allotted is to be computed. As in the probability program, T1, T2, T3, T4, T5, T6, ANORS, and the data for comparing average monthly demands with the demand-floor cutoff criteria are also part of the input. In this case, ANORS represents only the desired probability of spares sufficiency; whether this probability level can be attained will depend upon whether enough spares can be bought within the cost constraint. The flying-hour programs for desired columns are also inputted.

In COST 1, the proportionality factors for computing average demand for site spares, average demand per month for demand-floor consideration, and the average demand for back-up spares are calculated as in PROB1 (first part of probability-constraint program) for all of the items (either the consumables or the repairables, depending upon which constraint is being used). The multiplicity of bases to particular columns is determined — i.e., how many bases have a column-7 flying-hour program, how many have column 8, etc.

2. JOB STEP COST 2

COST 2 uses the average demands for spares computed in COST 1 to calculate optimal quantities of spares based on two considerations: the desired probability of spares sufficiency and the maximum dollars that can be used to purchase spares.

This program computes optimum quantities of spares in a process that essentially comprises successive probability-constraint operations. A probability-constraint run is made at the desired probability level (entered by the program user), and the total cost of the spares determined is compared with the cost constraint (again entered by the program user). If the cost is within one percent of the entered constraint, the program stops. If the cost exceeds the entered constraint, then a second probability run is executed, with the entered probability constraint decreased and the comparison of costs made again. This continues until a run is made that results in a cross-section of spares whose cost is less than the entered cost constraint. At this point the probability constraint used is increased by one-half the value of

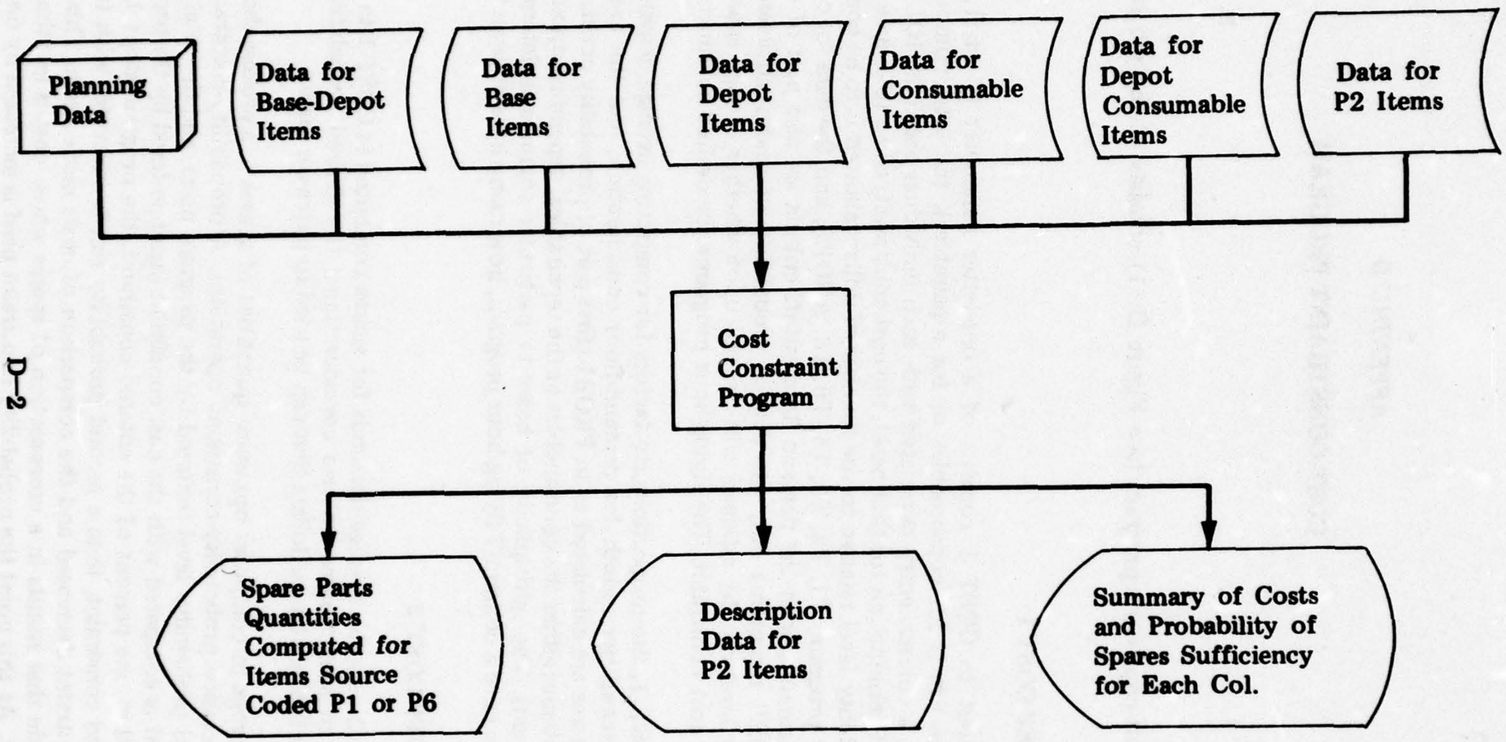


Figure D—1. **THROUGHPUT DESCRIPTION FOR COST-CONSTRAINT PROGRAM**

the previous decrease (example: go from 0.95 to 0.85, then to 0.90), and a run is made again and cost comparisons are made. This continues until the partitioning described above is less than a user-entered tolerance. (Warning! This tolerance must never be less than 0.01.)

Ultimately the program will stop with a cross-section of spares whose cost is within approximately 1 percent of the cost constraint. When the allowable tolerance is reached, if the total cost is greater than the constraint, the spare-parts quantities are considered optimal; if the total cost is not greater than the constraint, then the probability level is raised and spares are recomputed so that the total cost will be slightly greater than the constraint, hence giving an optimal solution.

After spare-parts quantities are computed, they are printed for each item together with description data exactly as in the probability program, except that in some cases where no sites are assigned to a particular column, 0's are printed. Descriptive data are printed for P2 items. This is followed by a summary that shows for each column and for the backup the total cost for each column and the probability of spares sufficiency. The overall cost of all spares is also indicated.

Program logic is shown in Figure D—2, which is followed by a complete program listing.

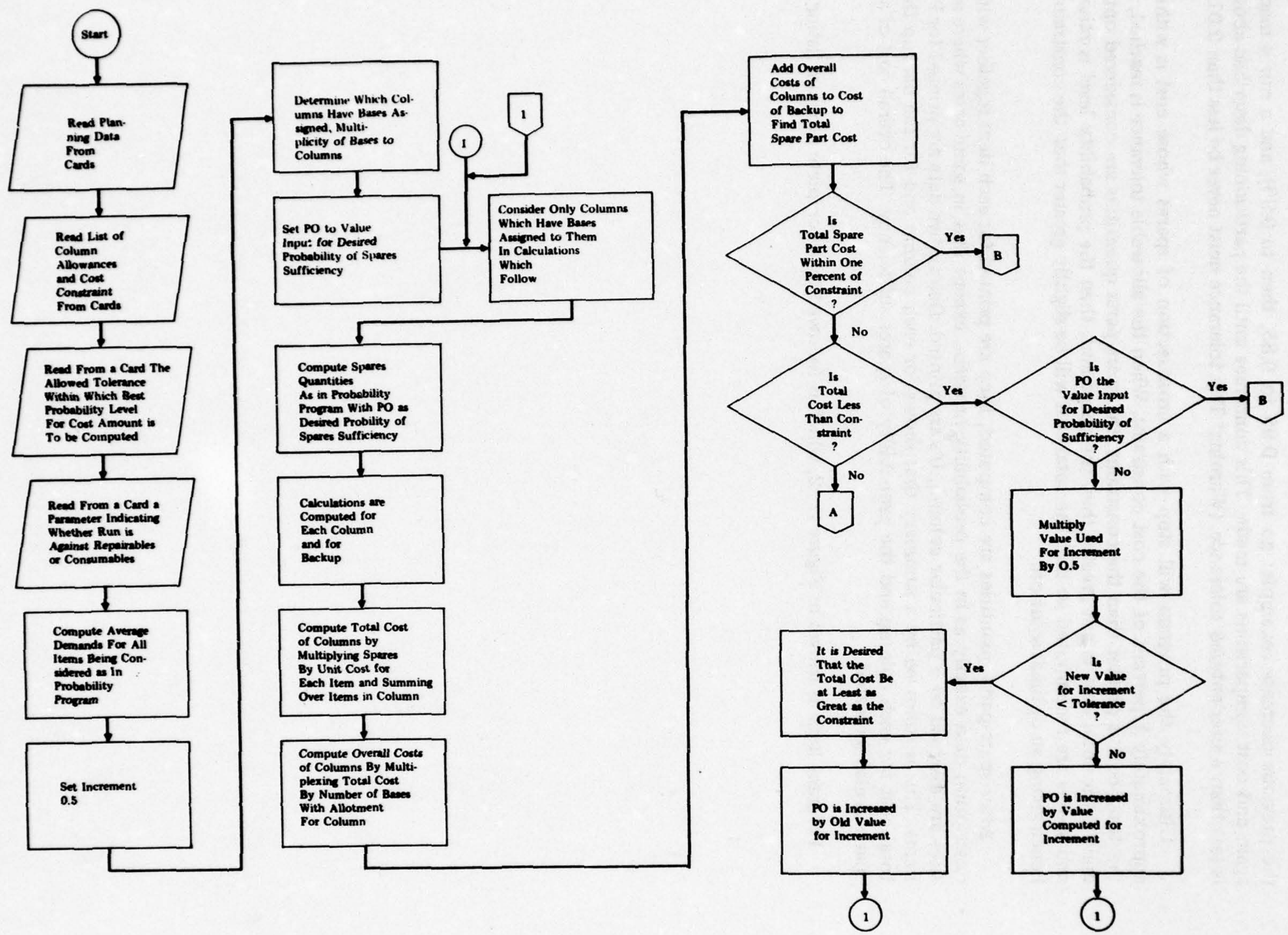


Figure D–2. FLOW CHART OF COST-CONSTRAINT PROGRAM

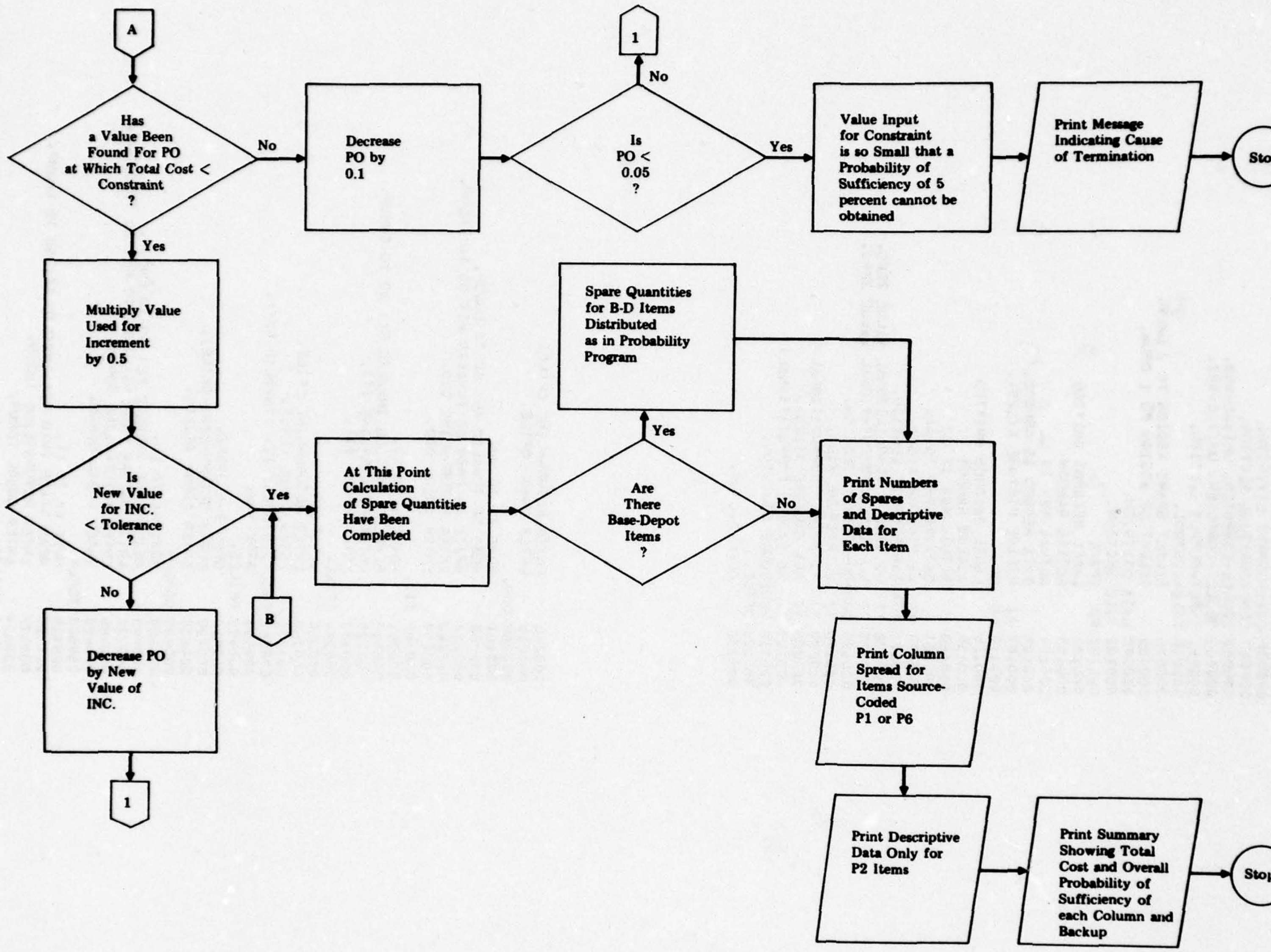


Figure D—2. (continued)

1COB HESV OFILE1

```
UNIVAC 490/491/492/494  COBOL COMPILATION    DATE-70050     TIME-00:00             VER6C

000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID. UNIVAC ASO.
000030 ENVIRONMENT DIVISION.
000040 CONFIGURATION SECTION.
000050 SOURCE-COMPUTER. UNIVAC-492.
000060 OBJECT-COMPUTER. UNIVAC-492.
000061 INPUT-OUTPUT SECTION.
000070 FILE-CONTROL.
000075      SELECT SPARE ASSIGN TO J DRUM.
000080      SELECT DF ASSIGN TO I DRUM.
000090 DATA DIVISION.
000100 FILE SECTION.
000101 FD  SPARE
000102      LABEL RECORDS OMITTED
000103      ACCESS RANDOM
000104      ACTUAL KEY IS LL
000105      DATA RECORD IS NSPARE.
000107 01   NSPARE PICTURE X(125).
000110 FD   DF
000130      LABEL RECORDS OMITTED
000150      ACCESS RANDOM
000160      ACTUAL KEY IS JJ
000161      DATA RECORD IS AA.
000170 01   AA PICTURE X(65).
000171 WORKING-STORAGE SECTION.
000172 77   JJ SIZE 5 COMPUTATIONAL VALUE ZERO.
000175 77   LL SIZE 5 COMPUTATIONAL VALUE ZERO.
000180 COMMON-STORAGE SECTION.
000190 77   BB PICTURE X(65).
000200 77   II SIZE 5 COMPUTATIONAL.
000205 77   NS1 PICTURE X(125).
000207 77   KK SIZE 5 COMPUTATIONAL.
000210 PROCEDURE DIVISION.
000220 OFILE.
000230      OPEN I-O DF.

000240      ENTER RETURN-LINE OFILE.
000250      ENTER COBOL OFILE.
000260 000.
000261      MOVE II TO JJ.
000265      READ DF INVALID KEY GO TO ENDUP.
000270      WRITE AA FROM BB INVALID KEY GO TO ENDUP.
000280      ENTER RETURN-LINE 000.
000290      ENTER COBOL 000.
000300 III.
000301      MOVE II TO JJ.
000310      READ DF INTO BB INVALID KEY GO TO ENDUP.
000320      ENTER RETURN-LINE III.
000330      ENTER COBOL III.
000340 CFILE.
000350      CLOSE DF.
000360      ENTER RETURN-LINE CFILE.
000370      ENTER COBOL CFILE.
000380 ENDUP. DISPLAY II, 'INVALID KEY'.
000390      STOP RUN.
000400 OFILE1.
000410      OPEN I-O SPARE.
000420      ENTER RETURN-LINE OFILE1.
000430      ENTER COBOL OFILE1.
000440 ODRUM.
000450      MOVE KK TO LL.
000460      READ SPARE INVALID KEY GO TO ENDUP.
000470      WRITE NSPARE FROM NS1 INVALID KEY GO TO ENDUP.
000480      ENTER RETURN-LINE ODRUM.
000490      ENTER COBOL ODRUM.
000500 IDRUM.
000510      MOVE KK TO LL.
000520      READ SPARE INTO NS1 INVALID KEY GO TO ENDUP.
000522      ENTER RETURN-LINE IDRUM.
000524      ENTER COBOL IDRUM.
000530 CFILE1.
000540      CLOSE SPARE.
000550      ENTER RETURN-LINE CFILE1.
000560      ENTER COBOL CFILE1.
```

```
@FOR YX CALVAL
         F O R T R A N   I V   C O M P I L A T I O N                          70050 SJ
          SUBROUTINE CALVAL(ITOT,ANORS,ANAC,CTOFA,NTYPE,JVAR)
           COMMON PRECL,SUMPC,COST,Y,VAL,II,CT,I
          DIMENSION CTOFA(5),NTYPE(5)
          IO=2
          IF(ITOT-JVAR)650,650,10
    10    P1=ANORS
           K = 1
   462     S = P1**(1./ITOT)
   475     DO 400 I =K,ITOT
           CALL III
           XL = Y*ANAC
          EXPDEM=CT*ANAC
          DO 300 J=1,5
          IF(I-NTYPE(J))301,301,300
   300    CONTINUE
   301    CTOF=CTOFA(J)
          IF(EXPDEM-CTOF)400,400,401
   401     IF (XL-30.) 405,400,400
   400     CONTINUE



           WRITE(IO,302)
   302     FORMAT(29H ALL ITEMS BELOW DEMAND FLOOR)
           STOP
   405      SUMPC = EXP(-XL)
            IREC = I
            PRECL = SUMPC
            II = 0
   420      IF (SUMPC-S) 410,415,415
   410      II = II+1
            PRECL = PRECL*XL/II
            IF (PRECL-1.E-35) 465,465,470
   465        K = I + 1
            GO TO 475
   470      SUMPC = SUMPC + PRECL
            GO TO 420
   415      VAL = (1.-SUMPC)/COST
   505      VALSAV = VAL
            CALL OOO
            PRODUC = SUMPC
            DO 425 I=1,ITOT
```

```
      IF (I-IREC) 430,425,430
430   CALL III
      EXPDEM=CT*ANAC
      DO 350 J=1,5
      IF(I-NTYPE(J))351,351,350
350   CONTINUE
351   CTOF=CTOFA(J)
      IF(EXPDEM-CTOF)429,429,432
429   II = 0
      GO TO 427
432   VAL = VALSAV * COST
      XL = Y*ANAC
      IF(XL-200.)500,416,416
416   II = XL + 1.645 * SQRT(XL) + .838253
427   SUMPC = 1.
      PRECL = 1.
      VAL = 0.
      GO TO 490
500   SUMPC = EXP(-XL)
      PRECL = SUMPC

      II = 0
445   IF (1.-SUMPC-VAL) 435,435,440
440   II = II+ 1
      PRECL = PRECL*XL/II
      IF (PRECL-1.E-35) 480,480,485
480   VAL = 0.
      SUMPC = 1.
      II = II-1
      GO TO 490
485   SUMPC = SUMPC + PRECL
      GO TO 445
435   PRODUC = PRODUC*SUMPC
      VAL = (1.-SUMPC)/COST
490   CALL OOO
425   CONTINUE
600   IF(PRODUC)650,650,601
601   IF(PRODUC-ANORS) 450,455,455
455   P1 = P1-.1
      IF (P1-.05) 650,460,460
460   F2 = VALSAV
```

```
      S = P1**(1./ITOT)
      I = IREC
      CALL III
      XL = Y*ANAC
520   IF (SUMPC-S)515,519,510
510   IF (II) 519,519,516
516   SUMPC = SUMPC-PRECL
      PRECL = PRECL*II/XL
      II = II-1
      GO TO 520
515   II = II + 1
      PRECL = PRECL*XL/II
      SUMPC = SUMPC + PRECL
519   VAL = (1.-SUMPC)/COST
      IF (F2-VAL) 605,455,605
605   VALSAV = VAL
      CALL OOO
      PRODUC = SUMPC
      DO 525 I=1,ITOT
      IF (I-IREC) 530,525,530



530   CALL III
      IF (II) 524,524,526
526   VAL = VALSAV*COST
      XL = Y*ANAC
      EXL = EXP(-XL)
      IF (EXL-1.E-35) 525,525,541
541   IF (PRECL-1.E-35) 532,532,545
532   II = 0
      SUMPC = EXL
      PRECL = EXL
245   IF (1.-SUMPC-VAL) 536,536,240
240   II = II + 1
      PRECL = PRECL*XL/II
      IF (PRECL-1.E-35) 280,280,285
280   VAL = 0.
      SUMPC = 1.
      II = II- 1
      GO TO 590
285   SUMPC = SUMPC + PRECL
      GO TO 245
```

```
545     IF (1.-SUMPC-VAL) 540,536,535
540     IF (II) 536,536,546
546     SUMPC = SUMPC - PRECL
        PRECL = PRECL*II/XL
        II = II-1
        GO TO 545
535     II = II + 1
        PRECL = PRECL*XL/II
        SUMPC = SUMPC + PRECL
536     VAL = (1.-SUMPC)/COST
524     PRODUC = PRODUC*SUMPC
590     CALL OOO
525     CONTINUE
        GO TO 600
650    ANORS1=ANORS
651    SUMLOG=0.
        DO 60I=1,ITOT
        CALL III
       DO 380 J=1,5
       IF  (I-NTYPE(J))  381,381,380


380    CONTINUE
381    CTOF=CTOFA(J)
       EXPDEM=CT*ANAC
       IF(EXPDEM-CTOF)6050,6050,382
6050    II=0
       SUMPC=1.
       VAL=0.
       GO TO 69
382     II = 0
       XL=ANAC*Y
       IF(XL-200.)700,70,70
70       II = XL+1.645*SQRT(XL)+.838253
         SUMPC = 1.
         PRECL = 1.
         VAL = 0.
         GO TO 69
700      SUMPC = EXP(-XL)
         PRECL = SUMPC
        VAL=(1.-SUMPC)/COST
         IF (SUMPC-ANORS1)61,69,69
```

```
61       II = II+1
         PRECL = PRECL*XL/II
         SUMPC = SUMPC + PRECL
         F2 = VAL
         VAL = (1.-SUMPC)/COST
        IF(F2-VAL)65,70,65
65       IF (SUMPC-ANORS1)61,69,69
69        CALL 000
 60     SUMLOG=SUMLOG+ALOG10(SUMPC)
        SUMLOG=SUMLOG+70.
        IF(SUMLOG)451,451,450
 451    ANORS1=ANORS1+.01
        IF(ANORS1-1.)651,652,652
 652    WRITE(IO,653)
 653    FORMAT($ UNDERFLOW WITH .99 PROB$)
        STOP
 450      RETURN
          END
#END
```

```
#FOR YX COSTA
        F O R T R A N   I V   C O M P I L A T I O N                70050 SJ
         COMMON PRECL,SUMPC,COST,Y,VAL,II,CT,I
          DIMENSION ANAC(10),  ISC(5),  RCDE(3),  ANOME(5),  PART(5),CTOF
        1   (5)
         DIMENSION MULT(10),NOMULT(10),ISC1(5)
          EQUIVALENCE(ISC(1),IBD),(ISC(2),IB),(ISC(3),ID),
        1    (ISC(4),IC),(ISC(5),IDC)
         INTEGER CMNT(2)
          INTEGER FSC,UNIT,RCDE,CSRC,     CMACC,ANOME,PART
         INTEGER FIIN(2),FSCM(2),CODM(2),APPL(2,10)
         INTEGER COL,R,C,RORC ,LRC,COGSYM,TWOV
         REAL MONTHS(5),MO
         DATA NZZZ/4HNZZZ/
         DATA R/1HR/,C/1HC/    ,TWOV/2H2V/
 C      READ LENGTHS OF EACH DATA AREA
         REWIND 9
         READ(9)ISC ,N6
         REWIND 9
         REWIND 10
         REWIND 11
```

```
      REWIND 12
      REWIND 13
      REWIND 14
      IO=2
      CALL OFILE
      REWIND 6
      REWIND 7
      NOCOL=0
      READ(1,601)RORC,DOLAR,NBASE,TOL
601   FORMAT(A1,9X,E12.5,8X,I4,6X,F6.5)
      DO 602  J=1,10
602   MULT (J)=0
      ONE PCT=.01*DOLAR
      DO 604 J=1,NBASE
      READ(1,603)COL
603   FORMAT(I2)
      COL=COL-6
604   MULT(COL)=MULT(COL)+1
       READ(1,5) T1,T2,T3,T4,T5,T6,ANORS
      READ(1,5)  (CTOF(I),MONTHS(I),I=1,5)



      READ(1,5)(ANAC(I),I=1,10)
  5   FORMAT(10E8.1)
      IF(RORC-R)201,202,201
201   IF(RORC-C)203,204,203
203   WRITE(IO,301)
301   FORMAT(10X,38HUNSPECIFIED REPAIRABLES OR CONSUMABLES)
      STOP
202   ISC1(4)=0
      ISC1(5)=0
      ISC1(1)=IBD
      ISC1(2)=IB
      ISC1(3)=ID
      ITOT=IBD+IB+ID
      JTOT=ITOT
      GO TO 210
204   ISC1(1)=0
      ISC1(2)=0
      ISC1(3)=0
      ISC1(4)=IC
      ISC1(5)=IDC
```

```
      ITOT=IC
      JTOT=IC+IDC
210   DO 216 J=1,10
      IF(MULT(J))215,216,215
215   NOCOL=NOCOL+1
      ANAC(NOCOL)=ANAC(J)
      NOMULT(NOCOL)=MULT(J)
216   CONTINUE
       IF (JTOT) 450,451,450
451    STOP
450    I = 0
      IDS=9
       DO 8 J=1,5
       IDS = IDS + 1
      NUMREC=ISC1(J)
       IF (NUMREC) 8,8,15
15    MO=MONTHS(J)
       DO 7 K=1,NUMREC
       I = I + 1
       READ (IDS) FIIN,IDD,FSC,NCC,FSCM,CODM,UNIT,RCDE,COST,PL,Z,
```

```
     1     APPL,IQU,CSRC,CMNT,CMACC,IPPQ,RRRQ,ANOME,PART,AAA,
     2     AAB,AAD,AAE,AAF,AMRF,RPF,     INSQTY ,LRC,COGSYM
       IF(COGSYM-TWOV)9,17,9
17     ISC1(J)=ISC1(J)-1
       IF(J-4)510,510,511
510    ITOT=ITOT-1
511    JTOT=JTOT-1
       I=I-1
       GO TO 7
9       CON1 = AAA * T3
        CON2 = AAB * T4
        CON3 = 3. * Z * PL * T1
C     IN THE CALCULATIONS WHICH FOLLOW Y IS A CONSTANT OF PROPORTIONALITY
C     FOR SITE SPARE EXPECTED DEMAND, CT IS A CONSTANT OF PROPORTIONALITY
C     FOR THE EXPECTED DEMAND CUTOFF CRITERION, ZZ IS THE EXPECTED DEMAND
C     FOR BACKUP SPARES
        GO TO (1001,1002,1003,1004,1005),J
C     BASE-DEPOT ITEM
1001   Y=(CON1+CON2)  /3000.
        ZZ=(AAF*CON3+AAD*(T2*T6/30.+CON3))/100.+RRRQ
```

```
      CT=(AAA+AAB)*MO/100.
C   RAT IS PERCENTAGE OF SITE SPARES ASSIGNED TO DEPOT FOR BASE-DEPOT
C   ITEMS.  THE OTHER SPARES ARE THE BASE PORTION
      RAT = CON2/(CON1 + CON2)
      WRITE(7) FIIN,COST,ZZ,RAT,ANOME,FSC,APPL,FSCM,CODM,UNIT,CSRC,
     1   CMNT,CMACC,AMRF,RPF,IQU,T3,RCDE,IPPQ,CT,NCC,PART,INSQTY,LRC
      GO TO 610
C   BASE ITEM
 1002 Y=CON1/3000.
      ZZ=AAF*CON3/100.+RRRQ
      CT=AAA*MO/100.
      GO TO 610
C    DEPOT ITEM
 1003 Y=CON2/3000.
      ZZ=AAD*(T2*T6/30.+CON3)/100.+RRRQ
      CT=AAB*MO/100.
      GO TO 610
C    CONSUMABLE ITEM
 1004 Y=AAB*T5/3000.
      CT=AAB*MO/100.

C   DEPOT CONSUMABLE
 1005 ZZ=AAD*PL*T1/33.333+AAE*PL/6.
610   IF(RCDE(1)-NZZZ)611,611,612
 611  ZZ=0.
 612  WRITE(6) FIIN,COST,ZZ,RAT,ANOME,FSC,APPL,FSCM,CODM,UNIT,CSRC,
     1    CMNT,CMACC,AMRF,RPF,IQU,    T3,RCDE,IPPQ,CT,NCC,PART,INSQTY,LRC
      IF(I-ITOT)2,2,7
  2   CALL OOO
7     CONTINUE
8      CONTINUE
      REWIND 6
      REWIND 7
      REWIND 10
      REWIND 11
      WRITE(11)ISC,ISC1,NOCOL,                              MULT,ITOT,
     1   JTOT,ONEPCT,DOLAR,CTOF,ANORS,TOL,N6
      WRITE(11)(ANAC(J),NOMULT(J),J=1,NOCOL)
      CALL CFILE
      STOP
       END
```

```
#FOR YX COSTB
          F O R T R A N   I V   C O M P I L A T I O N                        70050 SJ
      COMMON PRECL,SUMPC,COST,Y,VAL,II,CT,I,IIZ0(25),ISPARE
      DIMENSION IND(10)
       DIMENSION ANAC(10),  ISC(5),  RCDE(3),  ANOME(5),  PART(5),
     1    LLL(5),
     1   PBA(10),NSPARE(25,11),NS(11),MM(11),NUMBER(6)
      DIMENSION NMULT(10),MULT(10),TCOST(10),NS1(10),ANAC1(10),TCOST1(10
     1   ),PBA1(10),ISC1(5)
      DIMENSION AVAL(500),ASUMPC(500),APRECL(500),ACOST(500),AY(500),
     1   ISP(500),IPOINT(500),ACT(500)
      DIMENSION NTYPE(5),CTOF(5),CSAVE(5),CTOFF(6)
      DIMENSION YARRAY(200)
       EQUIVALENCE(ISC(1),IBD)
      REAL INCR
      INTEGER FIIN(2),FSCM(2),CODM(2),APPL(2,10)
      INTEGER CMNT(2)
       INTEGER FSC,UNIT,RCDE,CSRC,      CMACC,ANOME,PART ,LRC
      DATA IBLNK/1H /,IAST/1H*/
      READ(1,1492)JVARO
 1492  FORMAT(I5)





      IO=2
      CALL OFILE
      CALL OFILE1
      REWIND 5
      REWIND 6
      REWIND 7
      REWIND 8
      REWIND 11
      READ(11)ISC1,ISC,NBASE,MULT,ITOT,JTOT,ONEPCT,DOLAR,CTOF,ANORS1,
     1   TOL ,N6
       READ(11)(ANAC(J),NMULT(J),J=1,NBASE)
      REWIND 11
      NYNUM=ITOT/200
      IF(NYNUM*200-ITOT)1570,1571,1570
 1570 NYNUM=NYNUM+1
 1571 NYNUM1=NYNUM-1
      INCR=.1
      LESS=0
      ANORS=ANORS1
      K1=0
```

```
      DO 440 J=1,10
      IF(MULT(J))439,438,439
  438 ANAC1(J)=0
      GO TO 440
  439 K1=K1+1
      ANAC1(J)=ANAC(K1)
  440 CONTINUE
      NTYPE(1)=ISC(1)
      NTYPE(2)=ISC(2)+NTYPE(1)
      NTYPE(3)=ISC(3)+NTYPE(2)
      NTYPE(4)=ITOT
      NTYPE(5)=JTOT
      DO 301 J=1,NBASE
      TCOST(J)=0.
  301 PBA(J)=0.
       IF (JTOT) 450,451,450
  451  STOP
  450 ISPARE=0
       IF (ITOT) 580,580,449
  449  IZ3 = 0


  582  DO 10 J=1,NBASE
       ANACJ=ANAC(J)
       TCJ=0.
       IF(JVARO-ITOT)913,913,590
  590  JVAR=ITOT
       GO TO 915
  913  JVAR=JVARO
  915  CALL CALVAL(ITOT,ANORS,ANACJ,CTOF,NTYPE,JVAR)
        IOUT=0
       SS=1.
       DO 896 I=1,ITOT
       CALL III
  896  SS=SS*SUMPC
       IF(SS-ANORS)914,12,12
  914  DO 100 I=1,JVAR
        CALL III
       AVAL(I)=VAL
       ASUMPC(I)=SUMPC
       APRECL(I)=PRECL
       AY(I)=Y
```

```
      ACOST(I)=COST
      ISP(I)=II
      ACT(I)=CT
100   IPOINT(I)=I
      JVAR1=JVAR+1
      IF(JVAR1-ITOT)622,622,623
622   MIN=0
      DO 93 I=JVAR1,ITOT
      IF(MIN)494,493,494
493   AMIN=AVAL(1)
      J1=1
      DO 491 I1=2,JVAR
      IF(AVAL(I1)-AMIN)492,492,491
492   J1=I1
      AMIN=AVAL(I1)
491   CONTINUE
      MIN=1
494   CALL III
      IF(VAL-AMIN)93,93,94
94    AVAL(J1)=VAL



      ASUMPC(J1)=SUMPC
      APRECL(J1)=PRECL
      ACOST(J1)=COST
      AY(J1)=Y
      ACT(J1)=CT
      ISP(J1)=II
      IPOINT(J1)=I
      MIN=0
93    CONTINUE
      XMAX1=AVAL(1)
      JVAR1=1
      DO 96 I=2,JVAR
      IF(AVAL(I)-XMAX1)97,96,96
97    XMAX1=AVAL(I)
      JVAR1=I
96    CONTINUE
      GO TO 392
623   XMAX1=0.
      JVAR1=ITOT+1
392   IVAR2=JVAR1+1
```

```
      IVAR1=JVAR1-1
92    XMAX=XMAX1
      J1=0
      IF(IVAR1)191,193,191
191   DO 190 I=1,IVAR1
      IF(AVAL(I)-XMAX)190,192,192
192   XMAX=AVAL(I)
      J1=I
190   CONTINUE
193   IF(IVAR2-JVAR)196,196,197
196   DO 195 I=IVAR2,JVAR
      IF(AVAL(I)-XMAX)195,198,198
198   XMAX=AVAL(I)
      J1=I
195   CONTINUE
197   IF(J1)390,390,295
390   DO 391 K=1,JVAR
      I=IPOINT(K)
      PRECL=APRECL(K)
      SUMPC=ASUMPC(K)



      COST=ACOST(K)
      VAL=AVAL(K)
      CT=ACT(K)
      Y=AY(K)
      II=ISP(K)
391   CALL OOO
      IF(IOUT)914,914,12
295   IF(APRECL(J1)-1.E-35)323,323,322
323   SUMPC1=ASUMPC(J1)
      ASUMPC(J1)=1.
      AVAL(J1)=0.
      GO TO 261
322   ISP(J1)=ISP(J1)+1
      APRECL(J1)=APRECL(J1)*AY(J1)*ANACJ  /ISP(J1)
      SUMPC1=ASUMPC(J1)
      ASUMPC(J1)=ASUMPC(J1)+APRECL(J1)
      AVAL(J1)=(1.-ASUMPC(J1))/ACOST(J1)
      IF(XMAX-AVAL(J1))510,510,261
510   ISP(J1)=ISP(J1)-1
      GO TO 323
```

```
261   SS=SS+ASUMPC(J1)/SUMPC1
      IF(SS-ANORS)92,912,912
912   IOUT=1
      GO TO 390
12    I=0
      IF(IZ3)400,400,401
400   INUM=4
      GO TO 403
401    INUM = 5
403   DO 250 KKK=1,INUM
      IIZ2=ISC(KKK)
      IF(IIZ2)250,250,200
200   IF(IIZ2-25)201,201,202
201   IIREC=IIZ2
      GO TO 203
202   IIREC=25
203   DO 210 I5=1,IIREC
      I=I+1
      CALL III
      TCJ=TCJ+II*COST

210   IIZQ(I5)=II
      ISPARE=ISPARE+1
      CALL ODRUM
      IIZ2=IIZ2-IIREC
      IF(IIZ2)250,250,200
250   CONTINUE
      PBA(J)=SS
10    TCOST(J)=TCJ
      IF(IZ3)580,580,581
580   IZ3=1
      TEMP=PBA(1)
      ITEM1=ITOT
      ITOT=JTOT
      NTEMP=NBASE
      NBASE=1
       ASAVE = ANAC(1)
      ANAC(1)=1.
      TCTEMP=TCOST(1)
      DO 360 I=1,5
      CSAVE(I)=CTOF(I)
```

```
360  CTOF(I)=-10.
     REWIND 12
     IF(ITEM1)1500,1500,1501
1501  REWIND 12
     I=0
     DO 1572 IY=1,NYNUM
     IF(IY-NYNUM)1573, 1574,1573
1573  NRECY=200
     GO TO 1575
1574  NRECY=ITEM1-200*NYNUM1
1575  DO 910 NY=1,NRECY
     I=I+1
     CALL III
     YARRAY(NY)=Y
     READ(6)FIIN,COST,Y
910  CALL OOO
1572  WRITE(12)(YARRAY(NYS),NYS=1,NRECY)
1500  IF(ITEM1-ITOT)1503,1502,1502
1503  ITEM2=ITEM1+1
      DO 911  I=ITEM2,ITOT




      READ(6)FIIN,COST,Y
911  CALL OOO
1502  REWIND 12
      REWIND 6
      GO TO 582
581  PBU=PBA(1)
      PBA(1)=TEMP
      TCBU=TCOST(1)
      TCOST(1)=TCTEMP
      ITOT=ITEM1
      NBASE=NTEMP
       ANAC(1) = ASAVE
      DO 365 I=1,5
365  CTOF(I)=CSAVE(I)
      TTCOST=0.
      DO 110 I=1,NBASE
110  TTCOST=TTCOST+NMULT(I)*TCOST(I)
      TTCOST=TTCOST+TCBU
      DIFF=ABS(TTCOST-DOLAR)
      IF(DIFF-ONEPCT)1000,1000,112
```

```
112  IF(TTCOST-DOLAR)121,121,122
121  LESS=1
     IF(ANORS-ANORS1)127,1000,1000
127  INCR=INCR*.5
     ANORS=ANORS+INCR
     IF(INCR-TOL)128,128,405
128  ANORS=ANORS+INCR
     GO TO 405
122  IF(LESS)125,125,126
125  ANORS=ANORS-INCR
     IF(ANORS-.05)136,136,405
405  IF(ITOT)450,450,404
404  I=0
     DO 406 IY=1,NYNUM
     IF(IY-NYNUM)1583,1584,1583
1583 NRECY=200
     GO TO 1585
1584  NRECY=ITEM1-200*NYNUM1
     READ(12)(YARRAY(NY),NY=1,NRECY)
1585  DO 406 NY=1,NRECY




     I=I+1
     CALL III
      Y=YARRAY(NY)
406  CALL OOO
     GO TO 450
136  WRITE(IO,635)
635  FORMAT(10X,$THE AMOUNT OF FUNDS ENTERED AS A CONSTRAINT IS TOO SMA
     1LL AS NOT EVEN 5 PERCENT SUFFICIENCY OF SPARES CAN BE OBTAINED$)
     STOP
126  INCR=INCR*.5
     ANORS=ANORS-INCR
     IF(INCR-TOL)1000,1000,405
1000 CALL CFILE
     NB1=NBASE+1
     IIITOT=0
     DO 770 I=1,5
770  LLL(I)=0
      DO 775 I=1,5
     IF(ISC(I))775,775,776
776  JJ=ISC(I)/25
```

```
      IF(JJ*25-ISC(I))714,739,739
714   JJ=JJ+1
739   IF(I-4)41,41,725
 41   IIITOT=IIITOT+JJ
725   LLL(I)=JJ
775   CONTINUE
       DO 715 J=1,5
      IF(ISC(J))715,715,785
785   K=0
      IF(J-1)733,734,733
733   JJ=J-1
      DO 745 III=1,JJ
745   K=K+LLL(III)
734   IZ=LLL(J)
      IF(J-4)731,731,732
731   DO 755 LL=1,IZ
      IFACT=0
      K=K+1
      DO 753  LL1=1,NB1
    - ISPARE=K+IFACT*IIITOT




      IFACT=IFACT+1
      CALL IDRUM
      DO 753 IZ1=1,25
753   NSPARE(IZ1,LL1)=IIZG(IZ1)
      IF(LL-IZ)757,756,757
756   I25=ISC(J)-25*(LL-1)
      GO TO 797
757   I25=25
797   DO 755 IZ1=1,I25
755   WRITE(8     )(NSPARE(IZ1,LL1),LL1=1,NB1)
      GO TO 715
732   K1=NBASE*IIITOT
      DO 795 LL=1,IZ
      K=K+1
      ISPARE=K+K1
      CALL IDRUM
      IF(LL-IZ)781,782,782
781   I25=25
      GO TO 791
782   I25=ISC(J)-25*(LL-1)
```

```
791  DO 795 I=1,I25
795  WRITE(8     ) IIZ0(I  )
715  CONTINUE
     CALL CFILE1
     REWIND 11
     REWIND8
     IF(IBD)950,950,951
951  DO 920 J=1,IBD
     READ(8     )(NS(I),I=1,NB1)
      READ(7) FIIN,COST,ZZ,RAT
     MM(NB1)=NS(NB1)
     DO 923 I=1,NBASE
     XII=NS(I)*RAT
     MM(I)=XII
     IF(XII-MM(I))922,923,922
922  MM(I)=MM(I)+1
923  NS(I)=NS(I)-MM(I)
     WRITE(5     )(MM(I),I=1,NB1)
920  WRITE(11     )(NS(I),I=1,NB1)
     REWIND 5




     REWIND 7
     REWIND 11
950  NUMBER(1)=ISC(1)
     NUMBER(2)=ISC(2)
     NUMBER(3)=ISC(1)
     NUMBER(4)=ISC(3)
     NUMBER(5)=ISC(4)
     NUMBER(6)=ISC(5)
     CTOFF(1)=CTOF(1)
     CTOFF(2)=CTOF(2)
     CTOFF(3)=CTOF(1)
     CTOFF(4)=CTOF(3)
     CTOFF(5)=CTOF(4)
     CTOFF(6)=CTOF(5)
      DO 720 I=1,6
724  JJ=NUMBER(I)
      GO TO (761,762,763,764,765,766),I
761  WRITE(IO,30)
     WRITE(IO,71)
71   FORMAT(1H ,18HROTABLE POOL ITEMS,//,1H ,25HBASE-DEPOT (BASE PORTIO
```

```
     1N),/)
1998  NT=6
      NT1=11
      J2=2
      GO TO 80
 762  WRITE(IO,72)
72    FORMAT(1H0,15HBASE REPAIRABLE,/)
      J2=J2+1
      GO TO 800
 763  WRITE(IO,30)
      WRITE(IO,773)
 773  FORMAT(1H ,15HATTRITION ITEMS,//,1H ,26HBASE-DEPOT (DEPOT PORTION)
     1,/)
 722  NT=7
      NT1=5
      J2=2
      GO TO 80
 764  WRITE(IO,74)
74    FORMAT(1H0,16HDEPOT REPAIRABLE,/)
      J2=J2+1

      GO TO 800
 765  WRITE(IO,75)
75    FORMAT(1H0,15HBASE CONSUMABLE,/)
      J2=J2+1
      GO TO 800
 766  WRITE(IO,30)
      WRITE(IO,76)
76    FORMAT(1H ,19HSYSTEM STOCKS ITEMS,/,1H0,16HDEPOT CONSUMABLE,/)
      J2=2
800   NT=6
      NT1=8
 80   DEM=CTOFF(I)
       IF (JJ) 720,720,7000
7000  DO 721 J=1,JJ
       J2 = J2 + 1
       READ(NT) FIIN,COST,ZZ,RAT,ANOME,FSC,APPL,FSCM,CODM,UNIT,CSRC,
     1  CMNT,CMACC,AMRF,RPF,IQU,T3,RCDE,IPPQ,CT,NCC,PART,INSQTY,LRC
       IF(J2-12)729,729,726
 726  J2=1
      WRITE(IO,30)
```

```
30    FORMAT(1H1,2X,4HFIIN,5X,12HNOMENCLATURE,15X,3HFSN,13X,15HREFERENCE
     1 NO.  ,6X,             36HFSCM  MODEL   UNIT SRCE MAINT   MACC,/,1
     1H ,82X,29HCODE   ISSU CDE  CODE      CODE,/,1H ,9X,56HMRF      RPF
     1    QUAN/   UNIT PRICE   LRC  TAT      RULES,7X,59HPAR     BU    7
     1 8    9   10   11   12   13   14   15   16,/,1H ,26X,4HPROV,22X,3
     1    HIMA,5X,4HCODE,8X,3HPLQ/15H    APPLICATIONS,85X,8H INS QTY)
729  IF(I-5)150,150,151
150  READ(NT1     )(NS(K1),K1=1,NB1)
     DO 1150 K1=1,10
     IF(ANAC1(K1))1152,1152,1149
1149 IF(CT*ANAC1(K1)-DEM)1151,1151,1152
1151  IND(K1)=IAST
     GO TO 1150
1152  IND(K1)=IBLNK
1150  CONTINUE
     K2=0
     DO 1250 K1=1,10
     IF(MULT(K1))1248,1249,1248
1249  NS1(K1)=0
     GO TO 1250
1248  K2=K2+1
     NS1(K1)=NS(K2)
1250  CONTINUE
      WRITE(IO,33) FIIN,ANOME,FSC,FIIN,     PART,      FSCM,CODM,UNIT,
     1     CSRC,CMNT,CMACC,AMRF,RPF,IQU,COST,LRC,T3,RCDE,IPPQ,
     2   NS(NB1),(NS1(K1),IND(K1),K1=1,10)
33   FORMAT(1H ,A3,A4,2X,5A4,5X,A4,1H-,A3,1H-,A4,4X,5A4,2X,A4,A1,
     1  1X,A4,A3,2X,A2,3X,A2,4X,2A1,7X,A1/
     2          4X,2(3X,F7.3),I8,F12.2,2X,A5,F6.0,2X
     3    A4,A2,A4,1X,I5,1X,    I6,10(I4,A1)     )
     WRITE(IO,34)APPL,INSQTY
     GO TO 721
151  READ(8     )NS(NB1)
      WRITE(IO,33) FIIN,ANOME,FSC,     FIIN,PART,      FSCM,CODM,UNIT,
     1     CSRC,CMNT,CMACC,AMRF,RPF,IQU,COST,LRC,T3,RCDE,IPPQ,
     2  NS(NB1)
     WRITE(IO,34)APPL     ,INSQTY
34   FORMAT(1X,10(A4,A3,2X),8X,I7/)
721  CONTINUE
720  CONTINUE
```

```
      IF(N6)1560,1560,1550
C     PRINT P2 ITEMS
1550  REWIND 15
      WRITE(IO,30)
      WRITE(IO,79)
79    FORMAT(1H ,8HP2 ITEMS/)
      J2=2
      DO 1551 K1=1,N6
      READ(15) FIIN,IDC,FSC,NCC,FSCM,CODM,UNIT,RCDE,COST,PL,Z,APPL,IQU,
     1  CSRC,CMNT,CMACC,IPPQ,RRR,ANOME,PART,AAA,AAA,AAA,AAA,AAA,AMRF,
     1  RPF ,INSQTY,LRC
      J2=J2+1
      IF(J2-12)1553,1553,1552
1552  J2=1
      WRITE(IO,30)
1553  WRITE(IO,33) FIIN,ANOME,FSC,FIIN,    PART,     FSCM,CODM,UNIT,
     1    CSRC,CMNT,CMACC,AMRF,RPF,IQU,COST,LRC,T3,RCDE,IPPQ
1551  WRITE(IO,34)APPL,INSQTY
      REWIND 15
1560  K2=0



      DO 1330 K1=1,10
      IF(MULT(K1))1330,1330,1328
1328  K2=K2+1
      TCOST1(K1)=TCOST(K2)
      PBA1(K1)=PBA(K2)
1330  CONTINUE
      WRITE(IO,1340)TCBU
      WRITE(IO,1341)PBU,ANORS1
1340  FORMAT(1H1,10X,25HOVERALL COST OF BACKUP IS,14X,F12.0)
1341  FORMAT(11X,30HPROBABILITY OF SUFFICIENCY IS ,F5.3/13X,28HCOMPARED
     1 TO A CONSTRAINT OF ,F5.3)
      DO 1350 K1=1,10
      K2=K1+6
      IF(MULT(K1))1348,1348,1349
1348  WRITE(IO,1342)K2
1342  FORMAT(/11X,24HNO BASES ASSIGNED COLUMN,I3)
      GO TO 1350
1349  TC1=TCOST1(K1)*MULT(K1)
      WRITE(IO,1343)K2,TCOST1(K1),MULT(K1),TC1
1343  FORMAT(/11X,21HTOTAL COST OF COLUMN ,I2,4H IS ,F12.0/11X,28HNUMBER
```

```
      1 OF BASES ASSIGNED IS ,I2/11X,25HOVERALL COST OF COLUMN IS,14X,
      2  F12.0)
       WRITE(IO,1341)PBA1(K1),ANORS1
 1350  CONTINUE
       WRITE(IO,1345)TTCOST,DOLAR
 1345  FORMAT(//11X,25HTHE TOTAL OVERALL COST IS,12X,F14.0/13X,27HCOMPAR
      1ED TO A CONSTRAINT OF,8X,F14.0)
       WRITE(IO,1607)ANORS
 1607  FORMAT(//10X,$IF A PROBABILITY CONSTRAINT RUN IS DESIRED, COMPATI
      1BLE WITH THIS COST RUN, A CONSTRAINT OF $,F6.4,$ SHOULD BE USED$)
       REWIND 5
       REWIND 6
       REWIND 7
       REWIND 8
       STOP
       END
#END
#END

#FIN
```

Preceding Page Blank - No Filmed

APPENDIX E

EDIT PROGRAM

An edit program has been developed that will check for data errors in the input to the ARINC Research Spares-Optimization Program. This program (EDIT) will review the format of information for each item, eliminating items with essential information missing, and subsequently will group similar type items for input to the spares-optimization program. The inputs to EDIT are tapes prepared by the data-conversion program (IFINALTAPE, NUMBEROFFIINS), which convert MDF data in a UICP Input Data Transcript format to a form acceptable as input to the spares-optimization program. A general description is presented in Figure E—1.

For each item the following tests are made:

1. Is source code anything other than P1, P2, or P6?
2. Is item identification code a value greater than five?
3. Is cost a value less than or equal to zero?
4. For identification codes of one or two (consumable items) are the product of removals/hr × quantity/application × number of applications × percent/application and the product of overhaul replacement rate × quantity/application × percent/application × number of overhauls equal to zero?
5. For an identification code of three, is the product of removals to depot/hr × quantity/application × number of applications, as well as the RRR Quantity, equal to zero?
6. For an identification code of four, are the product of removals to base/hr × quantity/application × number of applications and the product of removals to depot/hr × quantity/application × number of applications and the RRR Quantity equal to zero?
7. For an identification code of five, is the product of removal rate to base/hr × quantity/application × number of applications, as well as the RRR Quantity, equal to zero?

If the answer to any of the above questions is yes, the item is rejected. If the percentage of items rejected becomes greater than a maximum value specified by the user, the program is terminated.

In addition to the tests described above, EDIT checks the value of production lead time; if this number is zero, a value read by the program at execution time is substituted for the zero.

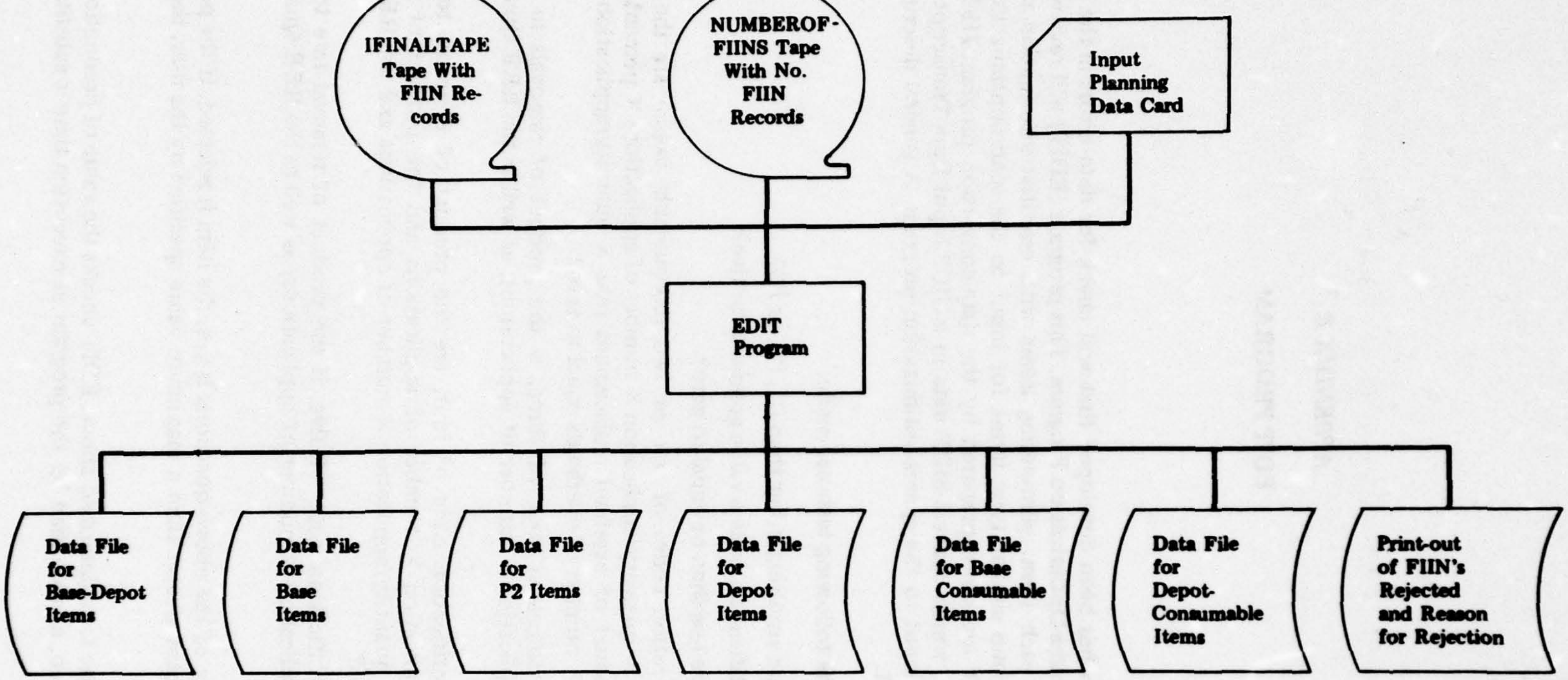


Figure E–1. **THROUGHPUT DESCRIPTION OF EDIT PROGRAM**

The input required by EDIT consists primarily of two tapes created by the data-conversion program that converts data from the UICP Input Data transcript format; the first tape contains a record for each FIIN; the second tape contains the number of FIIN records from the first. However, the user must also furnish a data card specifying the percentage of items rejected that cannot be exceeded and the value of production lead time to substitute for zero values of that number. An explanation of the input card is given in Table E—1.

Table E—1. **EDIT INPUT-CARD DESCRIPTION**

Card Columns	Format	Description
1-10	F10.2	Maximum percentage of items that can be rejected before program termination
11-20	F10.2	Value of production lead time to substitute for zero values of that number

After an item has been judged acceptable, it is stored in one of the five files that are inputs to the spares-optimization program. Each of these files contains items of a single type.

Each item is processed in turn until all items have been checked or until the percentage of items rejected becomes greater than the maximum specified, at which time the program will terminate.

The program logic is shown in flow-chart form in Figure E—2, which is followed by a complete program listing.

The EDIT program has been designed to aid the user in screening the input data and eliminating items that lack essential information, resulting in an optimization program that gives more meaningful results.

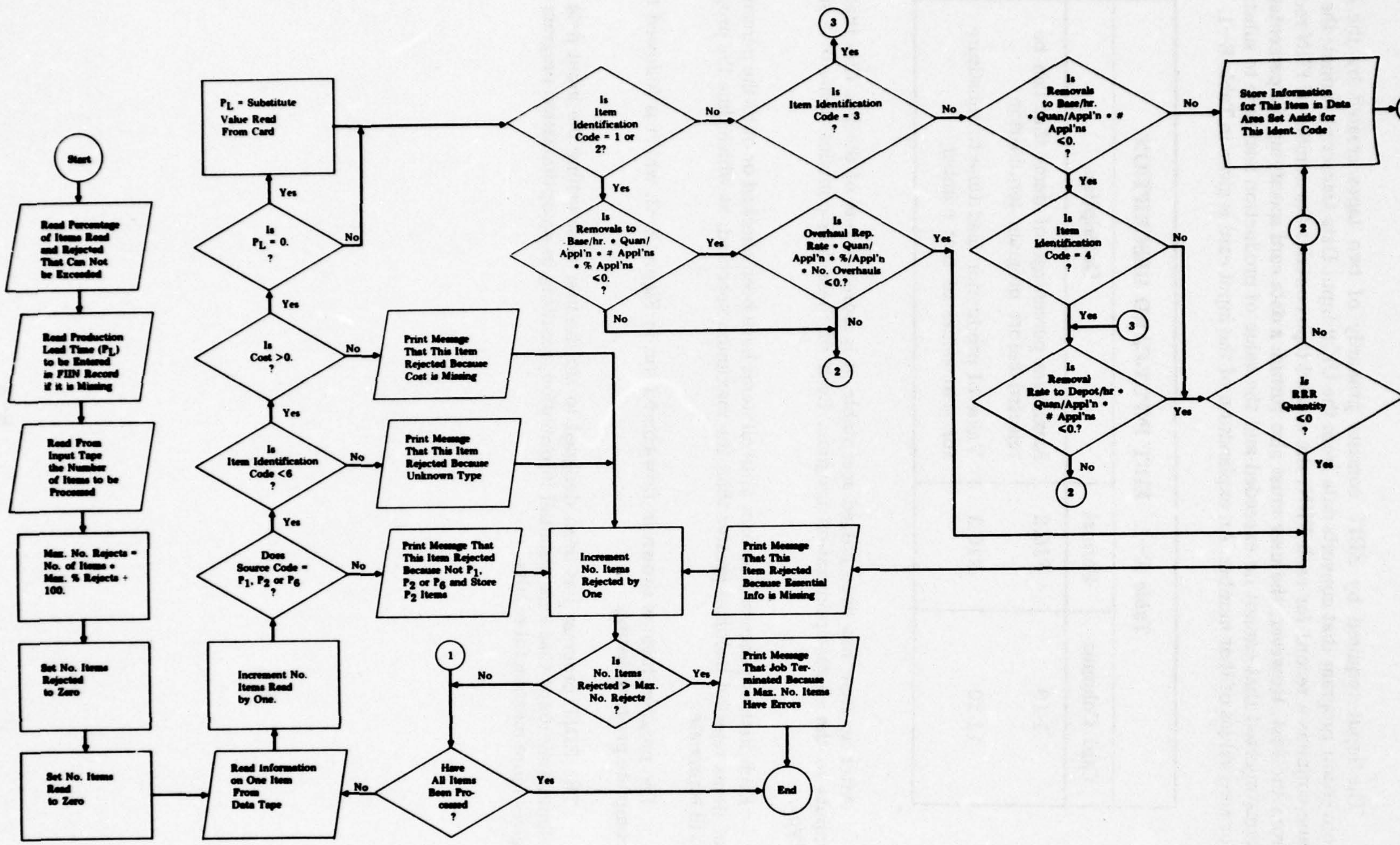


Figure E–2. FLOW CHART OF EDIT PROGRAM

```
#FOR YX EDIT1
      F O R T R A N   I V   C O M P I L A T I O N                    70050 SJ
C     EDIT PROGRAM FOR F. JACOBY WRITTEN BY D. EWELL, FEBRUARY 1970
      INTEGER FIIN,FSC,FSCM,CODE,UOI,RULES,APPL,SRCE,CMAINT(2),CMACC,
     1   XITEM,PNUM  ,LRC,COGSYM ,P1,P6 ,P2
      DIMENSION FIIN(2),  FSCM(2),  CODE(2),  RULES(3),  APPL(2,10),
     1    XITEM(5),  PNUM(5)
      DATA P1/2HP1/,P6/2HP6/,P2/2HP2/
      REWIND 4
      REWIND 5
      REWIND 15
      IO = 2
      WRITE(IO,999)
999   FORMAT(1H1)
C     READ PERCENTAGE OF RECORDS REJECTED BEFORE ABORT AND VALUE OF PL FOR
C                SUBSTITUTION IF BLANK
      READ(1,1000) APCT,SUBPL
1000  FORMAT(2F10.2)
C     READ NUMBER OF RECORDS ON TAPE FOR EDIT
      READ(5)NUMREC
      INUM = (IFIX(APCT*NUMREC)+50)/100




C     INITIALIZATION
      N1 = 0
      N2 = 0
      N3 = 0
      N4 = 0
      N5 = 0
      N6=0
      NR = 0
      NREJ = 0
C     READ RECORDS FROM TAPE
100   IF(NR-NUMREC)101,200,200
101   READ(4) FIIN,IDC,FSC,NCC,FSCM,CODE,UOI,RULES,COST,PL,Z,
     1         NQUAN,SRCE,CMAINT,CMACC,IPPO,
     2 RRR,(XITEM(I),PNUM(I),I=1,5),AAA,AAB,AAD,AAE,AAF,AMRF,RPF,PCT,
     3   INSQTY,APPL   ,LRC,COGSYM
      NR = NR + 1
      IF(SRCE-P1)125,115,125
125   IF(SRCE-P6)135,115,135
135   IF(SRCE-P2)145,155,145
115   IF (IDC-6) 5,10,10
```

```
5       IF (COST) 20,20,15
15      IF (PL) 30,25,30
25      PL = SUBPL
30      GO TO (40,40,50,60,70),IDC
C    IDC = 1 OR IDC = 2
 40    IF(AAD) 46,46,45
 46    IF(AAE)80,80,45
45      GO TO  (42,44), IDC
42      N5 = N5 + 1
        IPRT = 14
        GO TO 90
44      N4 = N4 + 1
        IPRT = 13
        GO TO 90
C    IDC = 3
 50    IF(AAB)56,56,55
 56    IF(RRR)80,80,55
55      N3 = N3 + 1
        IPRT = 12
        GO TO 90
C    IDC = 4
 60    IF(AAA)64,64,62
 64    IF(AAB)66,66,62
 66    IF(RRR)80,80,62
62      N1 = N1 + 1
        IPRT = 10
        GO TO 90
C    IDC = 5
 70    IF(AAA)76,76,75
 76    IF(RRR)80,80,75
75      N2 = N2 + 1
        IPRT = 11
90      WRITE(IPRT) FIIN,IDC,FSC,NCC,FSCM,CODE,UOI,RULES,COST,PL,Z,
     1     APPL,NQUAN,SRCE,CHAINT,CHACC,IPPB,
     2             RRR,    XITEM,PNUM,AAA,AAB,AAD,AAE,AAF,AMRF,RPF
     3    ,INSQTY ,LRC,COGSYM
        GO TO 100
C    ITEM IS P2
155    WRITE( 15 ) FIIN,IDC,FSC,NCC,FSCM,CODE,UOI,RULES,COST,PL,Z,
     1     APPL,NQUAN,SRCE,CHAINT,CHACC,IPPB,
```

```
     2              RRR,     XITEM,PNUM,AAA,AAB,AAD,AAE,AAF,AMRF,RPF
     3    ,INSQTY ,LRC,COGSYM
      N6=N6+1
      GO TO 100
C   SOURCE CODE IS NOT P1,P2, OR P6
 145  WRITE(IO,136)FIIN
 136  FORMAT(1H0,$THIS FIIN NO. $,A3,A4,$ CANNOT BE PROCESSED, ITEM NOT
     1P1,P2, OR P6$)
      GO TO 105
C    UNKNOWN ITEM
10      WRITE(IO,1002) FIIN
 1002 FORMAT(1H0,$THIS FIIN NO. $,   A3,A4,$ CANNOT BE PROCESSED, UNKNOW
     1N TYPES)
105     NREJ = NREJ + 1
        IF (NREJ-INUM) 100,110,110
C   NEGATIVE OR ZERO COST
20      WRITE(IO,1003) FIIN
1003    FORMAT(1H0,$THIS FIIN NO. $,  A3,A4,$ CANNOT BE PROCESSED, COST I
     1S MISSING$)
        GO TO 105
C    ESSENTIAL INFO MISSING
80      WRITE(IO,1004) FIIN
1004    FORMAT(1H0,$THIS FIIN NO. $,  A3,A4,$ CANNOT BE PROCESSED, ESSENT
     1IAL INFO IS MISSING SUCH AS MRF,RPF,RRR,QUAN$)
        GO TO 105
110     WRITE(IO,1005) APCT
1005    FORMAT(1H0,10X,F10.1,$ PER CENT OF ITEMS HAVE ERRORS. JOB IS TERM
     1INATED$)
        N1 = 0
        N2 = 0
        N3 = 0
        N4 = 0
        N5 = 0
 200  WRITE(9)N1,N2,N3,N4,N5 ,N6
        STOP
        END
!END
!LOAD NMY EDIT1,EDIT
!END
```

Preceding Page Blank - Not Filmed

APPENDIX F

ATTRIBUTES CALCULATED IN THE DATA-CONVERSION PROGRAM

Quantities calculated are:

$$AAA = \sum_{\text{appcodes}} (AI)_i * (QA)_i * (RPF)_i$$

$$AAB = \sum_{\text{appcodes}} (AI)_i * (QA)_i * (MRF)_i$$

$$AAD = \sum_{\text{appcodes}} (AI)_i * (QA)_i * (PA)_i * (MRF)_i$$

$$AAE = \sum_{\text{appcodes}} \begin{pmatrix} NHA \\ OHLS \end{pmatrix}_i * (PA)_i * (QA)_i * (OR)_i$$

$$AAF = \sum_{\text{appcodes}} (AI)_i * (PA)_i * (QA)_i * (RPF)_i$$

$$RRR\ Quan = \sum_{\text{appcodes}} \begin{pmatrix} NHA \\ OHLS \end{pmatrix}_i * (PA)_i * (QA)_i * RRR_i$$

NHA is Next Higher Assembly; OHLS is Overhauls.

The quantities calculated for specific item types are:

	AAA	AAB	AAD	AAE	AAF	RRR QUAN	Item Type Code
B-C		X	X	X			2
D-C			X	X			1
B-R	X				X	X	5
B/D-R	X	X	X		X	X	4
D-R		X	X			X	3

B-C is Base Consumable
D-C is Depot Consumable
B-R is Base Repairable
B/D-R is Base-Depot Repairable
D-R is Depot Repairable

PRECEDING PAGE BLANK - NOT FILMED

APPENDIX G

PROCEDURE FOR ADDING A DEN TO THE DATA-CONVERSION PROGRAM

An identification DEN can be incorporated in the ARINC Research Data-Conversion Program by adding it to the list of acceptable DENs in WRTFIN (the COBOL input/output routine), by processing it in DCPASS1, and by adding it to all input/output lists in DCPASS 1 and DCPASS2.

To follow an example through all phases of addition, DEN BØ54 is selected; the value in columns 20-28 is to be printed out under "new information" in the IOL-listing. In WRTFIN, the COBOL subroutine of DCPASS1, locate in the Working Storage section the level 01 array MAJ-ID-TBL. This table defines all DENS that will be passed along to DCPASS1 from the input tape. Add a level 03 filler with the value of the DEN. The following table, MAJ, must be increased by one to accommodate the new DEN.

In the Procedure Division, locate paragraph ANAL, which analyzes the input record for an acceptable DEN. The index should be incremented by 1 to reflect the added DEN. The COBOL subroutine is now complete.

In DCPASS1, a variable must be named to hold the new data. B54CST is selected. Since this is an implied real value (does not begin with I, J, K, L, M, or N), it need not be declared REAL. A variable for the DEN is chosen to be the same as its value, BØ54, for clarity. The variable BØ54 is to be compared with an integer variable, APPCD; therefore, BØ54 must be declared INTEGER.

Now, a value for BØ54 must be assigned in a DATA statement. The notation is DATA BØ54/4hBØ54/, which is explained as follows:

BØ54 (variable name)/(open description) 4 (characters) h (Hollerith) BØ54 (value)/ (close description).

The logic of checking DENs must now be followed to find where the B54CST data can be read. Since all non-DØØ9 DENs are read first, the correct spot must be between lines 103 and 200.

The arithmetic IF in FORTRAN gives three addresses to branch to if the result is negative, zero, or positive, in that order.

After line 103, we see such a branch. If the tested DEN is greater than BØØ2, we jump to line 107; otherwise, we jump to line 105 and read the local routing code. Since BØ54 is greater than BØØ2, we go to 107. Since BØ54 is greater than BØ53, we go to 130. At 130, we test BØ54 against CØØ4 and find the result negative, and go to 140. At 140, we test the DEN

against BØ67 and would expect to transfer to 102, since the test is negative. But we do not go to 102, since that would read a new card. Hence, we change the address of that jump to a new line number, 144, where we CONTINUE (a no-operation). Now we test our DEN BØ54 against the test value JDENT. If we find it equal, we want to read in the B54CST, or else jump to 102. The B54CST can be read under the FORMAT at 1012, which skips 19 spaces, then reads 9 columns with 2 decimal places. The resulting sequence is:

```
140  CONTINUE
     IF (JDENT-BØ67) 144, 142, 102

142  CONTINUE
     READ (IHOLD, 1014) RULSCD
     GO TO 102

144  CONTINUE
     IF (JDENT-BØ54) 102, 146, 102

146  READ (IHOLD, 1012) B54CST
     GO TO 102
```

We have read the B54CST; now we must pass it on. At the end of DCPASS1, we output the entire FIIN, unformatted. It is only necessary to add B54CST to the output list, immediately behind the local routing code. DCPASS1 is now complete.

DCPASS2 is somewhat easier, but more cards must be changed because of the subroutine structure of the program. All items are in COMMON, so we must add B54CST to the COMMON list *in each subroutine*, in addition to the main driver. It is not necessary to declare it REAL.

Wherever there is a READ or WRITE of an entire FIIN record, B54CST must be added at the end. The only position in which the driver is affected is immediately after line 62, where the item is found to be a consumable and saved in ITAP1R.

In the various subroutines, 10 READ or WRITE statements are affected, and B54CST must be added to each of them. DCPASS2 is then complete, and the new DENs passed to the optimization programs.

The procedure for adding a DEN is not complicated, as has been shown above, but the addition affects many sections of the program.

APPENDIX H

PROGRAM INITIALIZATION FOR ANALYSTS AND PROGRAMMERS AND CONTROL CARD LISTING

This appendix describes, from the analyst and programmer viewpoints (See Table H—1), the initialization to be supplied by the analyst and entered onto the data cards by the programmer.

Of special interest to the programmer is the following narrative, which describes drum-scratch-area assignment as a function of number of items to be handled. (This applies to probability and cost-constraint programs.)

Appendix I provides complete operator instructions.

The maximum quantity of drum storage required for each item is 24 words. There are two data sets on drum; one holds 13 words for each item and the other holds 11 words for each item.

The number of words in each of these sets is determined by a card of the form

ASG ~~b~~ J ~~b~~ RAN, file code, number of words (see Figure H—1)

where "file code" is I or J and "number of words" is an actual number. The program control card decks are currently set up with 100000 as the number of words.

$$\text{Therefore, at least } \frac{(100000)_8}{(13)_{10}} = (2520)_{10} \text{ items}$$

can be handled. By increasing the size of the data sets, more items can be handled. For example, if "number of words" is increased to 400,000, then at least 10,000 items can be handled. Of course, the total number of items that can be handled is a function of how large an area may be used for scratch on the drum.

Table H–1. **PROGRAM INITIALIZATION CHART**

Program	Data Entry	How Determined	Card Column
Data-Conversion Program – Pass 1	None		
Data-Conversion Program – Pass 2	$\frac{\text{Flying Hours}}{\text{Month Repairables}} = \text{T2}$	$\frac{\left[\text{RO* Maint Cycles} - (\text{RO-5}) \text{ Maint Cycles}\right]\left[\frac{\text{Hrs}}{\text{MC}}\right]}{5 \text{ (months)}}$	Card #1 1–10 (Implied 3 decimal places in columns 8, 9, 10)**
	Overhauls per Aircraft		11–20 (Implied 3 decimal places in columns 18, 19, 20)**
	Systems per Aircraft		21–30 (Implied 3 decimal places in columns 28, 29, 30)**
Probability Program – Prob 1	Number of desired Columns in IOL spread		Card #1 1–5 (right justified)
	$\frac{\text{Flying Hours}}{\text{Month Consumables}} = \text{T1}$	$\frac{\left(\text{RO Maint Cycles}\right)\left(\frac{\text{Hrs}}{\text{MC}}\right)}{\text{RO(months)}}$	Card #2 1–8***
	$\frac{\text{Flying Hours}}{\text{Month Repairables}} = \text{T2}$	$\frac{\left[\text{RO Maint Cycles} - (\text{RO-5}) \text{ Maint Cycles}\right]\left[\frac{\text{Hrs}}{\text{MC}}\right]}{5 \text{ (months)}}$	9–16***
	IMA TAT = T3		17–24***
	Resupply Time = T4		25–32***
	Protection Time (Consumables) = T5		33–40***
	Restockage Time = T6		41–48***
	Probability Constraint = ANORS		49–56***
Probability Program – Prob 1	Minimum Demand† Base Depot Item		Card #3 1–8
	Number of Months		9–16
	Minimum Demand Base Repairable Item		17–24
	Number of Months		25–32
	Minimum Demand Depot Repairable Item		33–40
	Number of Months		41–48
	Minimum Demand Base Consumable Item		49–56
	Number of Months		57–64
	Minimum Demand Depot Consumables		65–72
	Number of Months		73–80
Probability Program –Prob 1	Flying Hour Program†† Expressed as Flying Hours per Month Column 7		Card #4 1–8†††
	Column 8		9–16
	Column 9		17–24
	. Column 10		25–32
	. Column 11		33–40
	. Column 12		41–48
	. Column 13		49–56
	. Column 14		57–64
	. Column 15		65–72
	. Column 16		73–80
Probability Program –Prob 2	Number of items whose data can be kept in core simultaneously. (Large as possible)	500	1–5 (right justified)
Cost Program –Cost 1	An R or C	Whether run is to be against repairables or consumables	Card #1 1
	Cost Constraint		11–22 (must punch a decimal point)
	Number of Bases Considered		31–34 (right justified)
	Probability Tolerance		41–50 (must punch a decimal point)
Cost Program –Cost 1	For each base a card with the column selection for that base. (5 bases would require 5 cards)		1–2
	Next 3 cards are same as cards 2, 3, and 4 of Probability Constraint Program		
Cost Program –Cost 2	Input Card is same as to Probability Constraint—Prob 2.		

*RO = Requisitioning Objective

**A punched decimal point overrides the implied decimal point.

***A decimal point should be punched in each of these fields.

†Example: 1 demand in 6 months would require 1. to be entered in Field 1–8 and 6. to be entered in Field 9–16. If no demand floor is desired the minimum demand which should be entered is -1.

††If a column is not desired in IOL do not enter a Flying Hour Program.

†††Field should include a punched decimal point.

Figure H—1. **LISTING OF CONTROL CARDS USED TO EXECUTE THE PROGRAMS FROM AN OBJECT TAPE**

Data-Conversion Program

```
JOB   ARINC
ASG JWR TAPE,P,,ARINCOBJECT
IN R P,DCPASS1
ASG JWR TAPE,F,,INPUTTAPE
ASG JWR TAPE,H,,SCRATCH
ASG JWR TAPE,E,,SCRATCH
ASG J RAN,D,1000
ASG J RAN,I,10000
GO Y DCPASS1,JOB
END
FREE R H,NUMBEROFFIINS
FREE R F
FREE R E,FIINSTP
JOB   ARINC
ASG JWR TAPE,P,,ARINCOBJECT
IN R P,DCPASS2
ASG JR TAPE,F,,SCRATCH
ASG JR TAPE,D,,NUMBEROFFIINS
ASG JR TAPE,E,,FIINSTP
ASG JWR TAPE,K,,SCRATCH
ASG JR TAPE,G,,SCRATCH
ASG JR TAPE,H,,SCRATCH
ASG JR TAPE,I,,SCRATCH
ASG JR TAPE,J,,SCRATCH
GO Y DCPASS2,JOB
2560.       0.0        1.0
END
FREE R F,IFINALTAPE
FREE R D
FREE R E
FREE R P
END
FIN
```

Figure H—1. (continued)

Probability-Constraint Program

```
 JOB   ARINC
 ASG JWR TAPE,P,,ARINCOBJECT
 IN R P,EDIT,PROB1,PROB2
 ASG JR TAPE,D,,IFINALTAPE
 ASG JR TAPE,E,,NUMBEROFFIINS
 ASG JR TAPE,O,,SCRATCH
 ASG JR TAPE,M,,SCRATCH
 ASG JWR TAPE,N,,SCRATCH
 ASG J RAN,I,100000
 ASG J RAN,J,100000
 ASG J RAN,K,50000
 ASG J RAN,L,50000
 GO Y EDIT,JOB
        75.        10.
 END
 FREE R D
 FREE R E
 ASG JR TAPE,F,,SCRATCH
 ASG JWR TAPE,G,,SCRATCH
 GO Y PROB1,JOB
    5
1655.6   2560.    3.      15.      90.      75.      .95
-1.      6.       -1.     6.       -1.      6.       -1.      6.      -1.
280.     710.     1150.   1647.    2390.
 END
 ASG JR TAPE,E,,SCRATCH
 ASG JWR TAPE,H,,SCRATCH
 FREE  L
 GO Y PROB2,JOB
00500
 END
 FREE R E,MDFUPDATE
 FREE R P
 END
 FIN
```

Figure H—1. **(continued)**

Cost-Constraint Program

```
JOB   ARINC
ASG JWR TAPE,P,,ARINCOBJECT
IN R P,EDIT,COST1,COST2
ASG JR TAPE,D,,IFINALTAPE
ASG JR TAPE,E,,NUMBEROFFIINS
ASG JR TAPE,O,,SCRATCH
ASG JR TAPE,M,,SCRATCH
ASG JWR TAPE,N,,SCRATCH
ASG J RAN,I,100000
ASG J RAN,J,100000
ASG J RAN,K,50000
ASG J RAN,L,50000
GO Y EDIT,JOB
       75.         10.
END
FREE R D
FREE R E
ASG JR TAPE,F,,SCRATCH
ASG JWR TAPE,G,,SCRATCH
GO Y COST1,JOB
C          200000.                0010       .01
07
07
07
07
07
07
08
08
08
09
11
1655.6   2560.    3.       15.      90.      75.      .95
-1.      6.       -1.      6.       -1.      6.       -1.      6.       -1.      6.
260.     710.     1150.    1647.    2390.
 END
 ASG JR TAPE,E,,SCRATCH
 ASG JWR TAPE,H,,SCRATCH
 GO Y COST2,JOB
00500
 END
 FIN
```

Preceding Page Blank - Filmed

APPENDIX I

OPERATOR INSTRUCTIONS

Table I—1 describes the operator actions on the 490 system necessary for execution of the ARINC Research Corporation programs, assuming that REX (Real-Time Executive routine) is resident in memory for all processing.

Table I–1. OPERATOR INSTRUCTIONS

Operator Action	Console Response	Remarks
	INITIAL ACTION	
Load SEER3C2 ERRATA cards followed by first program to be executed into the 1ØØ4 Card Reader. Mount SEER3C2 tape. Type on console following command LD¤T¤_¤_¤7474700322¤22ØØØ¤BØⓈ Logical unit for tape drives. — Particular drive holding SEER3C2 tape. — Will vary between the U-494 and U-490. (Value shown is for U-490.)	REX MSG 443 1547 C 21547 77777 OP REX SEER 3 PØ3 ENTERED: ØØØØ	
Type on console: PS ¤3¤Ⓢ	REX II PØ3 Input CHØ3 PØ3 Output CHØ3 PØ3 Output CHØ3 PØ3 DØ1	This starts execution of program 3 (i.e., the SEER-3 Monitor)
Press START, CLEAR, FEED, and RUN buttons on the 1ØØ4 console.		This is standard sequence to initialize the 1ØØ4. Should get a TA3 on the 1ØØ4.
Type on console: DØ1¤Ⓢ		SEER-3 monitor has been instructed to read and process cards in the primary input stream. After typing this command, the cards are read in and the job executed.
IF DATA CONVERSION PROGRAM PASS 1 IS BEING USED		
	PØ3 MOUNT ARINCOBJECT C 11 U ØØ PØ3 DØ1	
Mount ARINCOBJECT Tape on Channel 11 Tape Drive Zero. Type on console: DØ1¤Ⓢ		
	PØ3 MOUNT INPUTTAPE C 11 U Ø1 PØ3 DØ1	
Mount INPUTTAPE Tape on Channel 11 Tape Drive One. Type on Console: DØ1¤Ⓢ		
	PØ3 MOUNT SCRATCH C 11 U Ø2 PØ3 DØ1	
Mount SCRATCH Tape on Channel 11 Tape Drive Two. Type on Console: DØ1¤Ⓢ		

IF DATA CONVERSION PROGRAM PASS 1 IS BEING USED		
	PØ3 MOUNT ARINCOBJECT C 11 U ØØ PØ3 DØ1	
Mount ARINCOBJECT Tape on Channel 11 Tape Drive Zero. Type on console: DØ1¤Ⓢ		
	PØ3 MOUNT INPUTTAPE C 11 U Ø1 PØ3 DØ1	
Mount INPUTTAPE Tape on Channel 11 Tape Drive One. Type on Console: DØ1¤Ⓢ		
	PØ3 MOUNT SCRATCH C 11 U Ø2 PØ3 DØ1	
Mount SCRATCH Tape on Channel 11 Tape Drive Two. Type on Console: DØ1¤Ⓢ		
	PØ3 MOUNT SCRATCH C 11 U Ø3 PØ3 DØ1	
Mount SCRATCH Tape on Channel 11 Tape Drive Three. Type on Console: DØ1¤Ⓢ		
	PØ3 MOUNT ################## ON C 11 U Ø1 PØ3 DØ1	
Type on Console: DØ1¤Ⓢ		
	PØ3 STOP PØ3 DISMOUNT FILE ON C 11 U Ø2* LABEL IT NUMBEROFFIINS PØ3 DISMOUNT FILE ON C 11 U Ø3* LABEL IT FIINSTP	
IF DATA CONVERSION PROGRAM PASS 2 IS BEING USED		
(If starting with Pass 2, Mount ARINCOBJECT Tape on Drive Zero)	PØ3 MOUNT ARINCOBJECT C 11 U ØØ PØ3 DØ1	This tape may already be mounted, if Pass 2 immediately follows Pass 1. If job is a continuation, just clear delay, otherwise, dismount ARINCOBJECT and save.
	PØ3 MOUNT SCRATCH C 11 U Ø1 PØ3 DØ1	Dismount tape labeled INPUTTAPE which may be presently located on this drive, (if Pass 1 was immediately preceding).
Mount SCRATCH Tape on Unit One. Type on Console: DØ1¤Ⓢ		
Mount appropriate tapes on appropriate drives if not already there.	PØ3 MOUNT NUMBEROFFIINS C 11 U Ø2 PØ3 MOUNT FIINSTP C 11 U Ø3 PØ3 MOUNT SCRATCH C 11 U Ø4 PØ3 MOUNT SCRATCH C 11 U Ø5 PØ3 MOUNT SCRATCH C 11 U Ø6 PØ3 MOUNT SCRATCH C 11 U Ø7 PØ3 MOUNT SCRATCH C 11 U Ø8 PØ3 DØ1	

*This is an intermediate checkpoint at which time job may be terminated if desired. If not, don't dismount tapes. Leave write ring in on NUMBEROFFIINS Tape. Remove ring from FIINSTP.

(continued)

Table I–1. (continued)

Operator Action	Console Response	Remarks
Type on Console: DØ1¤ Ⓢ	PØ3 STOP PØ3 DISMOUNT FILE ON C 11 U Ø1 LABEL IT IFINALTAPE PØ3 DISMOUNT FILE ON C 11 U Ø2 LABEL IT NUMBEROFFIINS REX-MSG 527 3 P = 25513 HR:	Also dismount FIINSTP from drive C 11 U Ø3, and ARINCOBJECT from Drive C 11 U ØØ and save.
IF PROBABILITY CONSTRAINT PROGRAM IS BEING USED		
	PØ3 MOUNT ARINCOBJECT C 11 U ØØ PØ3 DØ1	
Mount ARINCOBJECT Tape on Channel 11 Tape Drive Zero.		
Type on Console: DØ1¤	PØ3 MOUNT IFINALTAPE C 11 U Ø1 PØ3 MOUNT NUMBEROFFIINS C 11 U Ø2 PØ3 MOUNT SCRATCH C 11 U Ø3 PØ3 MOUNT SCRATCH C 11 U Ø4 PØ3 MOUNT SCRATCH C 11 U Ø5 PØ3 DØ1	
Mount Tapes on appropriate drives.		
Type on Console: DØ1¤ Ⓢ	PØ3 STOP	
Remove IFINALTAPE from drive. Mount SCRATCH tape.	PØ3 MOUNT SCRATCH C 11 U Ø1 RXM MSG 21ØØ3ØØ 74ØØØ3Ø12Ø	
Type on Console: DØ1¤ Ⓢ	RXM ADVISE, PØ3, CH11, DØ1	
Remove NUMBEROFFIINS tape from drive 2. Mount SCRATCH tape.	PØ3 MOUNT SCRATCH C 11 U Ø2 PØ3 DØ1 1ØØ5	
Type on Console: DØ1¤ Ⓢ	PØ3 STOP PØ3 MOUNT SCRATCH C 11 U Ø6 PØ3 MOUNT SCRATCH C 11 U Ø7 PØ3 DØ1 1807	
Mount SCRATCH tapes on drives 6 and 7.		
Type on Console: DØ1¤ Ⓢ	PØ3 STOP PØ3 DISMOUNT FILE ON C 11 U Ø6 LABEL IT MDFUPDATE	
Dismount tape on drive 6 and label it MDFUPDATE. Dismount and save ARINCOBJECT Tape.	RXM MSG 527 3 P = 25113 HR:	This indicates end of execution of the probability constraint program.
IF COST CONSTRAINT PROGRAM IS BEING USED		

Type on Console: DØ1□Ⓢ	RXM ADVISE, PØ3, CH11, DØ1	
Remove NUMBEROFFIINS tape from drive 2. Mount SCRATCH tape.	PØ3 MOUNT SCRATCH C 11 U Ø2 PØ3 DØ1 1ØØ5	
Type on Console: DØ1□Ⓢ		
	PØ3 STOP PØ3 MOUNT SCRATCH C 11 U Ø6 PØ3 MOUNT SCRATCH C 11 U Ø7 PØ3 DØ1 1807	
Mount SCRATCH tapes on drives 6 and 7.		
Type on Console: DØ1□Ⓢ		
	PØ3 STOP PØ3 DISMOUNT FILE ON C 11 U Ø6 LABEL IT MDFUPDATE	
Dismount tape on drive 6 and label it MDFUPDATE. Dismount and save ARINCOBJECT Tape.	RXM MSG 527 3 P = 25113 HR:	This indicates end of execution of the probability constraint program.
	IF COST CONSTRAINT PROGRAM IS BEING USED	
	PØ3 MOUNT ARINCOBJECT C 11 U ØØ PØ3 DØ1 1821	
Mount ARINCOBJECT Tape on Drive Zero.		
Type on Console: DØ1□Ⓢ	PØ3 MOUNT IFINALTAPE C 11 U Ø1 PØ3 MOUNT NUMBEROFFIINS C 11 U Ø2 PØ3 MOUNT SCRATCH C 11 U Ø3 PØ3 MOUNT SCRATCH C 11 U Ø4 PØ3 MOUNT SCRATCH C 11 U Ø5 PØ3 DØ1 1824	
Mount appropriate tapes on drives 1 through 5.		
Type on Console: DØ1□Ⓢ	PØ3 STOP PØ3 MOUNT SCRATCH C 11 U Ø1 RXM MSG 21ØØ3ØØ 74ØØØ3Ø12Ø RXM ADVISE, PØ3, CH11, D Ø1	
Remove IFINALTAPE from Drive 1 and mount a scratch tape.		
Type on Console: DØ1□Ⓢ	PØ3 MOUNT SCRATCH C 11 U Ø2 PØ3 DØ1 1828	
Remove NUMBEROFFIINS from Drive 2 and mount a scratch tape.		
Type on Console: DØ1□Ⓢ	PØ3 STOP PØ3 MOUNT SCRATCH C 11 U Ø6 PØ3 MOUNT SCRATCH C 11 U Ø7 PØ3 DØ1	
Mount SCRATCH tapes on drives 6 and 7.		
Type on Console: DØ1□Ⓢ	PØ3 STOP RXM MSG 527 3 P = 25113 HR:	This indicates end of cost constraint program.

2